UNIVERSITY OF TECHNOLOGY, SYDNEY

Gaussian Processes for Information-theoretic Robotic Mapping and Exploration

by

Maani Ghaffari Jadidi

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

at the
Centre for Autonomous Systems
Faculty of Engineering and Information Technology
University of Technology, Sydney

March 2017

Declaration of Authorship

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signed: _____

Date: _____

Gaussian Processes for Information-theoretic Robotic Mapping and Exploration

by

Maani Ghaffari Jadidi

Submitted to the Faculty of Engineering and Information Technology
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Abstract

This thesis proposes a framework for autonomous robotic mapping, exploration, and planning that uses Gaussian Processes (GPs) to model high-dimensional dense maps and solve the problem of infinite-horizon planning with imperfect state information.

Robotic exploration is traditionally implemented using occupancy grid representations and geometric targets known as frontiers. The occupancy grid representation relies on the assumption of independence between grid cells and ignores structural correlations present in the environment. We develop an incremental GP occupancy mapping technique that is computationally tractable for online map building and represents a continuous model of uncertainty over the map spatial coordinates. The standard way to represent geometric frontiers extracted from occupancy maps is to assign binary values to each grid cell. We extend this notion to novel probabilistic frontier maps computed efficiently using the gradient of the GP occupancy map and propose a mutual information-based greedy exploration technique built on that representation. A primary motivation is the fact that high-dimensional map inference requires fewer observations, leading to a faster map entropy reduction during exploration for map building scenarios.

The uncertainty from pose estimation is often ignored during current mapping strategies as the dense belief representation of occupancy maps makes the uncertainty propagation impractical.

Additionally, when kernel methods are applied, such maps tend to model structural shapes of the environment with excessive smoothness. We show how the incremental GP occupancy mapping technique can be extended to accept uncertain robot poses and mitigate the excessive smoothness problem using Warped Gaussian Processes. This approach can model non-Gaussian noise in the observation space and capture the possible non-linearity in that space better than standard GPs.

Finally, we develop a sampling-based information gathering planner, with an information-theoretic convergence, which allows dense belief representations. The planner takes the present uncertainty in state estimation into account and provides a general framework for robotic exploration in *a priori* unknown environments with an information-theoretic stopping criterion. The developed framework relaxes the need for any state or action space discretization and is a fully information-driven integrated navigation technique.

The developed framework can be applied to a large number of scenarios where the robot is tasked to perform exploration and information gathering simultaneously. The developed algorithms in this thesis are implemented and evaluated using simulated and experimental datasets and are publicly available as open source libraries.

Thesis Supervisors: Jaime Valls Miro and Gamini Dissanayake

Acknowledgements

First and foremost I would like to express my sincere gratitude to my supervisors, Prof. Jaime Valls Miro and Prof. Gamini Dissanayake, for being enthusiastic in guiding my research, for the freedom to learn and explore exciting topics, and for their patience. I hope I continue to collaborate with you in the future. I acknowledge the scholarships to support my research and make my study possible.

Besides my advisors, I would like to thank Dr. Teresa Vidal Calleja, Prof. Shoudong Huang, Prof. Juan Andrade Cetto, and Dr. Rafael Valencia for their guidance and friendship. I learned a lot from our weekly SLAM meetings organized by Shoudong and Teresa. Collaborating with Juan and Rafael during my first year was prolific.

I am deeply grateful to my external examiners, Professors Geoffrey Hollinger and Jon Kim, for reviewing this thesis, their comments, suggestions, and explanations.

My sincere thanks also goes to Prof. Kazuhiro Nakadai and Dr. Keisuke Nakamura, who provided me an opportunity to join their team as an intern at HRI-JP and experience living in Japan. During my six month visit, I made great friends and memories. I also thank Dr. Mitesh Patel for hosting me at FXPAL as an intern; I enjoyed working in a friendly environment and living in California.

I would like to thank my friends and colleagues at the Centre for Autonomous Systems for the numerous stimulating and helpful discussions and their friendship. It has been great living in beautiful Sydney and spending time with all of them. Special thanks to my dear friend Kasra Khosoussi who has been like a brother to me during the past five years. I guess we've also had the same number of fights I've had with my brothers back home! Thanks to my labmate Raphael Falque for always finding a way to be next to me, despite changing my desk two times plus moving to a new building. Over time, we have developed a non-verbal language that surprises others. Needless to say, the 2015 IROS together and three weeks traveling in Europe are among our memorable times.

Thanks to Lakshitha Dantanarayana AKA Lakipedia, for his friendship and vast knowledge on nearly anything available on the Internet. I thank my friends and labmates Gibson Hu, Andrew To, Buddhi Wijerathna, Mohammad Norouzi, Daobilige Su, Mahdi Hassan, Phillip Quin, Kanzhi

Wu, Liye Sun, Liang Zhao, Alen Alempijevic, Bradley Skinner, Teng Zhang, Peter Ward, Nalika Ulapane, Linh Van Nguyen, Asok Aravinda, Julien Collart, Alexander Virgona, Dinuka Abeywardena, Lasitha Piyathilaka, Leo Shi, James Poon, Brendan Emery, Katherine Waldron, Cédric Le Gentil, Karthick Thiyagarajan, David Valiente Garcia, and many others.

I would also like to thank my former supervisors, Prof. Morteza Mousakhani, Dr. Ehsan Hashemi, Prof. Alireza Mohammad Shahri, and Prof. Houman Sadjadian for their guidance and encouraging me to pursue my studies.

I thank my parents and my brothers, Navid and Houtan, for their unconditional love and care. I would not have made it this far without them.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

2D	Two-Dimensional. 2, 36
APS	Active Pose SLAM. 117, 118, 128, 129
ARD	Automatic Relevance Determination. 80
ASV	Autonomous Surface Vehicle. 110
AUC	the Area Under the receiver operating characteristic Curve. 43, 44, 79
BCM	Bayesian Committee Machine. 15, 42–45, 103
COM	Continuous Occupancy Map. 2, 35, 39, 41, 128, 130
EK	Expected Kernel. 79–81, 87
EKF	Extended Kalman Filter. 9, 11
ESM	Expected Sub-Map. 79–81, 87
GP	Gaussian Process. 2, 3, 5, 15, 16, 24, 26, 27, 32, 33, 35, 37–43, 49, 51, 70, 71, 73, 74, 77, 79, 87, 97, 103, 119–121, 125, 126, 129, 130
GPOM	Gaussian Processes Occupancy Map. 15, 16, 41, 49–52, 56, 60, 68, 72, 73, 79–81, 84, 86, 87, 125

GPVR Gaussian Processes Variance Reduction. 98, 103, 104, 111, 112, 114

I-GPOM Incremental Gaussian Processes Occupancy Map. 39, 41, 49, 72, 110, 117

IIG Incrementally-exploring Information Gathering. 89–91, 94–97, 102, 105, 109, 110, 114, 119, 121, 122, 124

JSD Jensen-Shannon Divergence. 60, 63

KLD Kullback-Leibler Divergence. 21, 22, 60

MDP Markov Decision Process. 52

MEU Maximum Expected Utility. 52

MI Mutual Information. 50, 51, 67, 97, 98, 101, 102, 111–114

MIUB Mutual Information Upper Bound. 102, 111–114

MSE Mean Squared Error. 128

NF Nearest Frontier. 128, 129

NLML Negative Log of the Marginal Likelihood. 25, 38, 39, 78, 79

OGM Occupancy Grid Map. 2, 10, 12, 13, 29, 30, 40, 56, 128

POMDP Partially Observable Markov Decision Process. 13, 52, 92

PSD Positive Semi-Definite. 26

RIC Relative Information Contribution. 96

RIG Rapidly-exploring Information Gathering. 89–92, 94–97, 110, 124

ROS Robot Operating System. 27

SE Squared Exponential. 26, 27, 37, 74, 80, 120

SLAM Simultaneous Localization And Mapping. 2–4, 8, 9, 12, 13, 29, 30, 69

UGPVR Uncertain Gaussian Processes Variance Reduction. 98, 104, 110–114, 124

WGP Warped Gaussian Process. 70, 87

WGPOM Warped Gaussian Processes Occupancy Map. 72, 73, 79–81, 84, 86, 87, 125

Nomenclature

t	Time step
$\approx$	Approximately equal
$\sim$	Distributed according to
$\triangleq$	Definition
$\varnothing$	Empty set
$\backslash$	Matrix left division
SE(2)	2D special Euclidean group
SO(2)	2D special orthogonal group
$R(\cdot)$	Rotation matrix
$\lvert\cdot\rvert$	Absolute value
a_t	Action at time step t
b_{occ}	Belief for the occupied map point
b_{free}	Belief for the unoccupied map point
$\mathcal{O}(\cdot)$	Big O notation
$\lambda^{[i]}$	Bounded information associated with location i
b_t	Budget at time step t
b	Budget

l	Characteristic length-scale
y_+	Class label for occupied points
y_-	Class label for unoccupied points
$f_c(\cdot)$	Cost function
$k(\cdot,\cdot)$	Covariance function of Gaussian processes
$\boldsymbol{K}$	Covariance matrix of Gaussian processes
$\mathbb{Cov}[\cdot]$	Covariance of random variables/vectors
$\lvert\cdot\rvert$	Determinant of a matrix
$h(\cdot)$	Differential entropy of a random variable
$H(\cdot)$	Entropy of a random variable
$\lVert\cdot\rVert$	Euclidean norm
$\mathbb{E}[\cdot]$	Expected value of a random variable
$\tilde{k}(\cdot,\cdot)$	Expected/Modified covariance function
$\mathcal{X}_f$	Free workspace
$\Gamma(\cdot)$	Gamma function
H_n	Hermite polynomials
$\mathcal{H}$	Hypothesis
$\boldsymbol{I}_n$	Identity matrix of size n
$f_I(\cdot)$	Information function
α	Information gain factor
$\boldsymbol{f}_*$	Latent variable of Gaussian processes
$\int(\cdot)$	Lebesgue integral

γ	Logistic regression classifier weight
$m^{[i]}$	Map occupancy status at location i
h_{sat}	Map saturation entropy
p_{sat}	Map saturation probability
δ_{map}	Map spatial resolution
$\boldsymbol{X}_f^{[k]}$	Matrix of sampled unoccupied points along the k-th sensor beam
$\max(\cdot)$	Maximal element of a set
$f_m(\cdot)$	Mean function of Gaussian processes
$\boldsymbol{z}_t^{[k]}$	Measurement at time step t by the k-th sensor beam
$\boldsymbol{z}_t$	Measurement at time step t
$\alpha_t^{[k]}$	Measurement bearing at time step t by the k-th sensor beam
$r_t^{[k]}$	Measurement range at time step t by the k-th sensor beam
$\boldsymbol{z}_{1:t}$	Measurements up to time step t
$\min(\cdot)$	Minimal element of a set
$K_\nu(\cdot)$	Modified Bessel function of the second kind of order ν
$\hat{I}(\cdot;\cdot)$	Mutual information approximation between two random variables
$I(\cdot;\cdot)$	Mutual information between two random variables
$\log(\cdot)$	Natural logarithm
n_i	Number of inducing points
n_p	Number of map points in the current perception field of the robot
n_m	Number of map points
n_q	Number of query points

n_z	Number of range-finder sensor beams per observation
n_s	Number of samples
n_f	Number of unoccupied sampled points
$h(\cdot,\cdot)$	Observation model
$\boldsymbol{x}_o^{[k]}$	Observed occupied point by the k-th sensor beam
$\boldsymbol{x}_o^{[k,i]}$	Observed occupied point's i-th dimension by the k-th sensor beam
β	Occupied boundaries factor
p_o	Occupied probability threshold
$a_t^\star$	Optimal action at time step t
$\mathcal{P}_t^\star$	Optimal trajectory at time step t
I_{RIC}	Penalized relative information contribution
$\mathcal{I}_t^{[k]}$	Perception field of the k-th sensor beam at any robot location at time step t
$\mathcal{G}$	Planning graph
T	Planning horizon
$\mathcal{T}$	Planning tree
$\sigma_{X_i\|Z}$	Posterior marginal variance of random variable X_i after taking observation Z
σ_{X_i}	Prior marginal variance of random variable X_i
$p(\cdot)$	Probability measure
$\boldsymbol{x}_*$	Query point
Z_t	Random variable for measurement at time step t
M	Random variable for the occupancy map
S	Random variable for the state

δ_{RIC}	Relative information contribution threshold
s_z	Resolution of the numerical integration
$\boldsymbol{Q}$	Robot motion noise covariance
$\boldsymbol{x}_t$	Robot pose at time step t
$\boldsymbol{x}_{1:t}$	Robot trajectory up to time step t
$\mathbb{Z}$	Set of integers
$\mathbb{N}$	Set of natural numbers
$\mathbb{R}_{\geq 0}$	Set of non-negative real numbers
$\mathcal{A}_t$	Set of possible actions at time step t
$\mathcal{A}$	Set of possible actions
$\mathcal{M}$	Set of possible occupancy maps
$\mathcal{Z}$	Set of possible range measurements
$\mathcal{S}$	Set of possible states
$\mathbb{R}$	Set of real numbers
$\mathcal{D}$	Set of training data
$\prod(\cdot)$	Set product
$\sum(\cdot)$	Set summation
$\operatorname{sgn}(\cdot)$	Sign function
σ_f^2	Signal variance
n_t	Size of training data
$u(\cdot)$	Total utility function
$\operatorname{tr}(\cdot)$	Trace of a matrix

$\mathcal{P}_t$	Trajectory at time step t
$\mathcal{P}$	Trajectory
$(\cdot)^T$	Transpose of a matrix
p_f	Unoccupied probability threshold
$\mathbb{V}[\cdot]$	Variance of a random variable
σ_n^2	Variance of the observation noise
$\boldsymbol{\psi}$	Vector of hyperparameters of a warping function
$\boldsymbol{\theta}$	Vector of hyperparameters
$\boldsymbol{t}$	Vector of latent targets
$\boldsymbol{y}$	Vector of targets
$g_w(\cdot)$	Warping function
$\mathcal{X}$	Workspace

Chapter 1

Introduction

The problem of a mobile robot navigating in an unknown environment unsupervised is the basic yet non-trivial goal of autonomous robotic research. Perception, localization, map building, planning and decision making are all challenging steps towards building a successful autonomous system. In addition to these steps, large-scale environments, long-term autonomy, adaptive decision making and system reliability are other concerns. The environment representation can also be a major challenge. Working on, for example, continuous, dense and discrete belief representations can restrict possible ways to approach the problem, both from theoretical and practical points of view.

Exploration in unknown environments is among the major challenges for an autonomous robot. Its numerous applications from search and rescue operations to space exploration programs highlight the importance of the problem. Exploratory problems mean both future state and measurements are unknown; consequently, in such problems it is non-trivial to say what is the optimal action. Exploring an unknown environment without any prior knowledge give rise to difficulties for the robot to make sequential decisions that maximize the long-term expected reward or "information gain". Among these difficulties, available information in the current state of the robot is limited to its perception field and the partially known state of its trajectory and the map as *a priori*. Therefore, there is always a contrary concept about the best possible decision at each sequence of time, i.e. *exploration* vs. *exploitation*. A choice which seems to be the best in the known part of a map may lead to an inefficient exploration policy in the global sense. This leads

the problem towards the sequential decision making under imperfect state information which is known to be NP-hard (Singh et al. 2009). The main motivation of this thesis is to find a practical non-myopic solution for the explained problems that delegates a higher level of autonomy for robotic systems.

1.1 Outline of the Problem and Assumptions

In this thesis, using continuous occupancy mapping and sampling-based robotic information gathering concepts, a framework for incremental robotic mapping and exploration of unknown environments where the robot pose and the environment are partially observable is presented.

Gaussian Processes (GPs) (Rasmussen and Williams 2006) are a modern mathematical tool for high-dimensional regression which can learn the affinity of observations through their provided covariance and mean functions. The method is known as continuous occupancy mapping (O'Callaghan et al. 2009; T O'Callaghan and Ramos 2012) in the literature as it places a joint normal distribution over query points and resulting maps are smooth surfaces, as opposed to classical Occupancy Grid Maps (OGMs) (Moravec and Elfes 1985; Elfes 1987). The continuous occupancy mapping technique is adopted for robotic exploratory purposes. This aim has been achieved by extension of the method to a computationally tractable incremental map building and computing frontier and information gain surfaces from an inferred Continuous Occupancy Map (COM).

Robotic information gathering uses a sampling-based planning strategy to calculate the cost and information gain while searching for traversable paths in the entire state space. This approach offers a solely information-driven robot control without specifying targets which distinguishes the approach from those in the literature. Furthermore, two major drawbacks of many previous works in the literature such as discretization of the action space and the maximum likelihood observations assumption are mitigated.

While the Simultaneous Localization And Mapping (SLAM) module can accept diverse types of sensor modalities, in this work, range-finder sensors which are the most common form of measurements in the literature are adopted. The environment is assumed to be Two-Dimensional (2D) with no specular effect. The robot operates on an even terrain and is equipped with laser

range-finder and odometric sensors. The environment is initially unknown to the robot, i.e. there is no prior information available. All the observations are locally made by the robot and there is no communication with any external source of information. It is assumed the sensor model is unbiased which, in the limit, guarantees convergence to the true values for the developed algorithms. However, perceived sensor data includes uncertainty which needs to be dealt with. SLAM as the first stage of a navigation module receives measurements and produces required robot poses. Therefore, dense map building and planning processes which use localization data are subject to inputs with uncertainties.

Initially, this study develops the incremental continuous occupancy mapping technique with known poses. We extend the method to frontier-based and information gain-based greedy exploration techniques. Afterward, we establish the required basis for occupancy mapping under pose uncertainty which is demonstrated using GPs with uncertain inputs. Eventually, we develop a sampling-based information gathering technique that provides a solution to the problem of robotic exploration in unknown environments with the partially observable state. The solution for the problem of robotic information gathering is also suitable for applications such as environmental monitoring tasks.

The evaluation of the proposed methods is demonstrated through simulation and experimental results in indoor and outdoor environments, together with publicly available datasets. Where possible, exhaustive comparisons have been made with state-of-the-art methods to study advantages and disadvantages of different algorithms.

1.2 Contributions

The main contributions of this thesis are as follows:

– A framework for Gaussian processes occupancy mapping using range-finder sensors is developed. The method is incremental and runs in nearly constant time for each update with an accuracy close to batch computation.

- A novel probabilistic geometric frontier representations by exploiting continuous occupancy mapping and using L^1-norm of the gradient of continuous occupancy maps is developed.

- A tractable technique for greedy information gain-based exploration is developed that takes into account all possible future observations and is based on the computation of the mutual information surface. The technique can deal with sparse measurements and uses a forward sensor model for map predictions.

- At this point, the robot pose uncertainty is incorporated into our incremental mapping framework. Gaussian processes occupancy mapping under uncertain robot poses is studied. We develop approaches which rely on uncertainty propagation through numerical integration over kernels. We also develop the expected sub-map fusion technique, a novel method for our incremental mapping framework. The proposed method uses local observations for map building and fuses the local and global maps by sampling from the marginal posterior distribution of the robot pose.

- A sampling-based planning algorithm for incremental robotic information gathering with an information-theoretic convergence criterion using the notion of penalized relative information contribution is developed. We formulate the problem in the form of an information maximization problem subject to a budget and the current state estimate constraints. We also prove an information-theoretic automatic stopping criterion for the entire mission based on the least upper bound of the average map entropy.

- The results from indoor and outdoor publicly available datasets are provided and compared with relevant methods.

- Open source implementation of the proposed algorithms is provided; this is available on:
  `https://github.com/MaaniGhaffari/gp_occ_mapping`
  `https://github.com/MaaniGhaffari/sampling_based_planners`

1.3 Thesis Overview

This thesis is organized in 8 chapters and structured as follows:

– In Chapter 2, the autonomous robotic navigation concept and its terminology are explained. The chapter includes reviews of the substantial research areas and the state of the art. We start with SLAM as the first step and continue with the literature review of exploration and planning problems. Active SLAM is also discussed as an important research branch and then the chapter is concluded with locating this thesis within the current state of autonomous robotic navigation research.

– In Chapter 3, we make clear the mathematical notation throughout this thesis. The required background knowledge about probability theory, information theory, numerical integration, Gaussian processes, and useful kernels are briefly explained. The chapter also includes basic techniques such as occupancy grid mapping and Pose SLAM. These techniques are employed in the development of the proposed algorithms or are part of the compared state-of-the-art techniques.

– In Chapter 4, the notion of continuous occupancy mapping is discussed and developed. The problem formulation of high-dimensional mapping is presented and is solved using Gaussian processes with batch and incremental approaches. We develop a Gaussian processes occupancy mapping method that allows for employing the map in practice and performing exploratory experiments.

– In Chapter 5, the geometric frontier concept is expanded through a continuous occupancy mapping method to a probabilistic frontier map representation. We develop a frontier-based exploration method capable of handling sparse observations and dense map entropy reduction exploiting continuous occupancy maps and numerical calculation of the mutual information of the map and future measurements. We present the comparison of classical and information gain-based exploration methods in diverse environments and scenarios.

– In Chapter 6, we study GP occupancy mapping problem extensions. The first problem is related to mapping with pose uncertainty and how to propagate pose estimation uncertainty into the map inference. In the second extension, we develop warped Gaussian processes continuous occupancy maps which can model the complexity in the structural shape of an environment with more accuracy by allowing non-Gaussian noise in the observation space.

– In Chapter 7, based on the concept of robotic information gathering, we develop a sampling-based incremental information gathering with, in overall, five information functions with the possibility to incorporate the robot pose and map uncertainty in the planning process. We also prove an information-theoretic termination condition for the entire exploration mission. Furthermore, we provide the demonstration of the developed algorithms in robotic explorations and environmental monitoring scenarios.

– In the last chapter, Chapter 8, we summarize and conclude this thesis by reviewing the proposed techniques, encountered challenges, and ideas for future work.

1.4 Publications

The work on the Gaussian processes mapping and exploration method was presented first in an RSS workshop (Ghaffari Jadidi et al. 2013a) and partially published in the 2013 Australasian Conference on Robotics and Automation (Ghaffari Jadidi et al. 2013b) and the 2014 IEEE International Conference on Robotics and Automation (Ghaffari Jadidi et al. 2014). The development of the mutual information-based exploration is presented in the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (Ghaffari Jadidi et al. 2015). The work on warped Gaussian processes mapping with uncertain inputs is published in the IEEE Robotics and Automation Letters (Ghaffari Jadidi et al. 2017) and accepted for presentation in 2017 IEEE International Conference on Robotics and Automation. The development of the Gaussian processes probability maps from visual features contributed to the publication in Valiente et al. (2015). The list of publications are as follows:

Journal Papers

1. **M. Ghaffari Jadidi**, J. Valls Miro, G. Dissanayake. *Sampling-based Incremental Information Gathering with Applications to Robotic Exploration and Environmental Monitoring*. Under review, The International Journal of Robotics Research.

2. **M. Ghaffari Jadidi**, J. Valls Miro, G. Dissanayake. *Warped Gaussian Processes Occupancy Mapping with Uncertain Inputs*. IEEE Robotics and Automation Letters. doi:10.1109/LRA.2017.2651154.

3. **M. Ghaffari Jadidi**, J. Valls Miro, G. Dissanayake. *Gaussian Process Autonomous Mapping and Exploration for Range Sensing Mobile Robots*. Conditionally accepted, Autonomous Robots.

4. D. Valiente, **M. Ghaffari Jadidi**, J. Valls Miro, A. Gil, O. Reinoso. *Information-based View Initialization in Visual SLAM with a Single Omnidirectional Camera*. Robotics and Autonomous Systems, Volume 72, 2015, pp. 93-104.

Conference Papers

5. **M. Ghaffari Jadidi**, J. Valls Miro, G. Dissanayake. *Mutual Information-based Exploration on Continuous Occupancy Maps*. In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 6086-6092.

6. **M. Ghaffari Jadidi**, J. Valls Miro, R. Valencia, J. Andrade-Cetto. *Exploration on Continuous Gaussian Process Frontier Maps*. In Proc. IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 6077-6082.

7. **M. Ghaffari Jadidi**, J. Valls Miro, R. Valencia, J. Andrade-Cetto, and G. Dissanayake. *Exploration using Information-based Reaction-diffusion Process*. In Australasian Conference on Robotics and Automation (ACRA), 2013.

8. **M. Ghaffari Jadidi**, J. Valls Miro, R. Valencia, J. Andrade-Cetto and G. Dissanayake. *Exploration in Information Distribution Maps*. Robotics: Science and Systems Workshop on Robotic Exploration, Monitoring, and Information Content, Berlin, Germany, 2013.

Chapter 2

Related Work

In this chapter, an introduction to autonomous robotic navigation and its related work is given. The problem is broken down into the major research areas at the present time. We review current work related to this thesis and locate this research within the state of the art.

2.1 Simultaneous Localization and Mapping

The problem in which a robot simultaneously builds a representation (map) of an unknown environment and estimates its pose in that map using relative measurements is known as SLAM. There have been notable efforts to solve the SLAM problem in the robotic community, and now there are several techniques that can provide a solution to this problem (Durrant-Whyte and Bailey 2006; Cadena et al. 2016). However, in GPS-denied unknown environments, solving SLAM practically is a key step for an autonomous mobile robot. Probabilistic SLAM is formulated as a *Bayesian estimation* problem in which the goal is to estimate the robot pose or trajectory and the feature map of the environment using noisy observations and control inputs. We can divide SLAM into online SLAM and full SLAM categories. In online SLAM, using available data up to the current time, the current robot pose and the map of the environment are estimated with respect to the *filtering distribution* of the robot pose and the map. Full SLAM methods estimate the *smoothing distribution* of the robot trajectory and the map given all available data.

Application of the Extended Kalman Filter (EKF) for solving SLAM is one of the first solutions to this problem. EKF-SLAM is an online SLAM method (Dissanayake et al. 2001) and has been studied extensively. There are two major drawbacks that are associated with this approach; first, the computational cost is cubic in the number of features due to the updating step of its dense covariance matrix for any new observations. Second, the linearization error in the EKF algorithm can result in overconfident and inconsistent estimates (Bailey et al. 2006a; Huang and Dissanayake 2007). In order to improve the performance and scalability of the EKF-SLAM, the aforementioned problems are partially addressed through building local maps and sub-map joining (Bosse et al. 2004; Estrada et al. 2005; Guivant and Nebot 2003; Paz et al. 2008).

The FastSLAM algorithm, introduced in Montemerlo et al. (2002), exploits Rao-Blackwellized particle filters (Doucet et al. 2000) and does not resort to linear Gaussian assumptions in EKF-SLAM. The key property of the Fast-SLAM relies on the fact that, given the robot trajectory, landmarks are conditionally independent. In practice, FastSLAM can generate accurate maps (Montemerlo et al. 2003), however, it suffers from degeneracy and depletion problems (Bailey et al. 2006b; Smith et al. 2013).

Later in Thrun et al. (2004), it was discovered that by avoiding marginalizing previous robot poses, as done in the online SLAM approach, the *information matrix* (the inverse covariance matrix) becomes exactly sparse [1]. This observation led to extensive studies of full SLAM using probabilistic graphical models representation and the *maximum a posteriori* estimate by solving a non-linear least squares problem (Dellaert and Kaess 2006; Grisetti et al. 2008; Kaess et al. 2011, 2008; Konolige et al. 2010; Sibley et al. 2008, 2009; Strasdat et al. 2010; Thrun and Montemerlo 2006).

The discussed graph-based SLAM is the feature-based form of SLAM, meaning the constructed graph includes all robot poses and landmarks. A recent popular formulation of SLAM is called *pose graph*, a variant of SLAM in which only the robot trajectory is estimated through the observation of relative constraints between robot poses. The pose-pose constraints can be obtained using scan matching (Diosi and Kleeman 2005; Borrmann et al. 2008; Olson 2009; Bosse and Zlot 2009) or visual place recognition techniques (Newman and Ho 2005; Ho and Newman 2007; Cummins and Newman 2008) between arbitrary poses and are called loop-closures.

[1] In the limit, the landmark estimates become fully correlated (Dissanayake et al. 2001), resulting in a heavy computational burden on EKF-SLAM.

In robotics, full SLAM using probabilistic graphical models representation and the maximum a posteriori estimate derived by solving a nonlinear least squares problem can be considered the state-of-the-art and we thus focus our attention on using a pose-graph estimation of the robot trajectory as in Ila et al. (2010); Valencia et al. (2013). Pose SLAM algorithm (Ila et al. 2010) is an information-based pose graph technique that considers the relative distance between poses and the expected information gain for each link (edge) to remove redundant poses and maintain highly informative loop-closures. The algorithm is capable of operating over long trajectories and therefore is suitable for experiments in large maps; however, the computational cost is mainly shifted to the data association process and the required parameters need to be tuned according to the size and shape of the environment.

2.2 Robotic Exploration and Motion Planning

An environment can be explored by directing a robot to *frontiers* that indicate unknown regions of the environment in the neighborhood of known free areas (Yamauchi 1997). Traditional autonomous exploration strategies have been devised to use OGMs (Moravec and Elfes 1985; Elfes 1987) to represent free, occupied and unknown regions. In Juliá et al. (2012), seven methods for autonomous exploration and mapping of unknown environments are studied and compared on the basis of exploration time and mapping error, both for single and cooperative mapping. The seven major approaches analyzed are nearest frontier (Yamauchi 1998), cost-utility (González-Banos and Latombe 2002), behavior-based coordinated (Lau 2003), coordinated (Burgard et al. 2000), market-based coordinated (Zlot et al. 2002), integrated (Makarenko et al. 2002), and hybrid integrated (Juliá et al. 2010). The comparison yielded conclusive results on the strategy to be used depending on the type of scenario and number of robots used. All the compared techniques used occupancy grids for map representation. In the following, we explain each technique briefly.

In Yamauchi (1998), the technique is based on the selection of the shortest path to the nearest frontier in which the only cost taken into account is traverse distance to a frontier cell. Therefore, there is no measure to evaluate the usefulness of a frontier cell and, moreover, this method does not include any coordination mechanism. In other words, in multi-robot scenarios robots which are in a close neighborhood may choose the same frontier. Practically, the method consists of the following steps. First, clustering the frontier map to identify neighborhood frontier cells.

By removing clusters smaller than a desired threshold, the centroids of the remaining clusters assemble a set of candidate goals. Next, Dijkstra's algorithm (Dijkstra 1959) can be employed to find the nearest frontier and the shortest path to it. The navigation cost defined by this algorithm is based on a Euclidean distance between cells from the current pose to all candidate goals. To ensure safe paths an infinite cost for obstacles and unknown cells can be set. Also, a penalizing factor for cells in the vicinity of obstacles would result in safer paths. Finally, by *backtracking*, the minimum cost path is achievable.

Cost-utility method (González-Banos and Latombe 2002) proposes an information gain measure that evaluates the utility of reaching a given cell. Frontier cells are nominated as candidate targets and the utility to reach a candidate cell is evaluated according to a *cost-utility* relation. The determination of candidate targets is done in the same way as the nearest frontier method, and Dijkstra's algorithm can be used to obtain the cost of each candidate. The behavior-based coordinated technique (Lau 2003) consists of three reactive behaviors in order to control the exploratory behavior of the robot. The behaviors are: `go to frontiers`, `avoid obstacles` and `avoid other robots`. In this technique, the potential associated with each behavior is modeled as a sum of Gaussians. Based on the linear combination of these behaviors, a global potential field is set up. Finally, the reverse direction of the gradient of the global potential field provides a direction for navigation. In order to recover from a local minimum in the potential field, the technique plans a path to the nearest frontier. The technique presented in Burgard et al. (2000) is a coordinated version of the nearest frontier method. The technique chooses a target using a function of the cost and the distance to candidate frontiers. Similarly, the determination of candidate targets is done in the same way as the nearest frontier method, and Dijkstra's algorithm can be used to obtain the cost of each candidate.

In Zlot et al. (2002), a market economy strategy is suggested to negotiate for applying some pre-defined rules. This approach employs the same utility function from the cost-utility model. Similar to the other techniques, candidate frontier cells are determined using clustering of frontier cells and Dijkstra's algorithm can be used to obtain the cost of each candidate. The proposed method in Makarenko et al. (2002) considers a localizability term in the utility function. The localizability function is calculated using an EKF and is normalized by the current uncertainty of the robot. The frontier cell that maximizes the utility function is selected as the destination cell. The determination of candidate targets is done in the same way as the nearest frontier

method and Dijkstra's algorithm can be used to obtain the cost of each candidate. Hybrid integrated algorithm (Juliá et al. 2010) consists of a hybrid exploration planning architecture. In this method, five reactive behaviors integrate the low-level planning. The behaviors are the three behaviors included in the behavior-based technique and two additional behaviors: `go to unexplored zones` and `go to precise poses`.

An alternative strategy to compute exploration paths is to treat frontiers as boundary conditions, and the explored area as a scalar potential field. The objective is to find "optimal" paths to the unexplored area. For instance, in Prestes e Silva et al. (2002), the technique computes the gradient of the potential field from solving Laplace's partial differential equation with Dirichlet boundary conditions (setting 1 for obstacles and 0 for free cells). The gradient descent direction indicates the path to the unexplored regions. In order to move beyond simple grid structures, in Shade and Newman (2011), the use of OctoMaps (Wurm et al. 2010; Hornung et al. 2013) is suggested. "Optimal" paths to the boundaries of unexplored regions are computed using the steepest descent on the associated gradient field. Another effort to cope with varying resolutions for the explored and unexplored regions in grid maps is presented in Shen et al. (2012). In this technique, registered sensor data is used to populate an occupied voxel, but a sparse free space representation is generated by a particle set. The evolution of a stochastic differential equation simulates the expansion of the system of particles in free space using Newtonian dynamics, and determines sparse unexplored regions.

Active Strategies

The following works fall into the area where *active perception* (Bajcsy 1988) concept is used to take actions that reduce the uncertainty of state variables. A combined information utility for exploration is developed in Bourgault et al. (2002) using the information-based cost function in Feder et al. (1999) and an OGM. A one-step look-ahead strategy is used to generate the locally optimal control action. The reported results indicated that the utility for mapping attracts the robot to unknown areas whilst the localization utility keeps the robot well localized relative to known features in the map. In Makarenko et al. (2002) an integrated exploration approach for a robot navigating in an unknown environment populated with beacons is proposed. This method examines the exploration problem through a total utility function consisting of the weighted sum

of the OGM entropy, navigation cost, and localizability. The vehicle pose is assumed to follow a Gaussian distribution, thus the entropy is proportional to the determinant of the vehicle pose covariance. In order to enhance the map quality of the EKF-based SLAM, an A-optimal criterion for autonomous exploration is examined in Sim and Roy (2005). To simplify the objective function, the map covariance matrix is approximated by ignoring the correlation among features. Furthermore, the search space of trajectories is limited by employing a breadth-first search algorithm to find global trajectories leading to a tractable method.

In Stachniss et al. (2005), Rao-Blackwellized particle filters are employed to compute map and robot pose posteriors. The proposed uncertainty reduction approach is based on the joint entropy minimization of the SLAM posterior. The information gain is approximated using ray-casting for a given action. In a similar framework in Carlone et al. (2010), the problem of active SLAM and exploration, and specifically the inconsistency in the filter due to the information loss for a given policy using the relative entropy concept is addressed. In Amigoni and Caglioti (2010), it is assumed that all random variables are normally distributed and an exploration strategy based on relative entropy metric, combined traveling cost, and expected information gain is proposed.

The techniques in Valencia et al. (2012); Vallvé and Andrade-Cetto (2013, 2014, 2015) evaluate exploratory and place revisiting paths, which are selected based on entropy reduction estimates of both map and path. Whilst the map entropy is computed on an occupancy grid at coarse resolution, path entropy is the outcome of Pose SLAM (Ila et al. 2010; Valencia et al. 2013), a delayed-state SLAM algorithm from the pose graph family. Given the inherent complexity in the formulation to calculate the joint entropy of robot pose and map, it is assumed that they are conditionally independent.

In Kim and Eustice (2015), a greedy approach for active visual SLAM that considers area coverage and navigation uncertainty is proposed. In Julian et al. (2014), the MI surface between a map and future measurements is computed numerically. The work assumes known robot poses, and relies on an OGM representation and measurements from a laser range-finder. The algorithm integrates over an information gain function with an inverse sensor model at its core. It is formally proven that any controller tasked to maximize an MI reward function is eventually attracted to unexplored areas.

Planning Under Uncertainty

Motion planning algorithms (Latombe 1991; LaValle 2006) construct a broad area of research in the robotic community. Here, we briefly summarize some of the fundamental and most relevant works related to this thesis. Under the presence of uncertainty, where the state is not fully observable, measurements and actions are stochastic. The sequential decision-making under uncertainty, in the most general form, can be formulated as a Partially Observable Markov Decision Process (POMDP) or optimal control with imperfect state information (Astrom 1965; Smallwood and Sondik 1973; Bertsekas 1995; Kaelbling et al. 1998). Unfortunately, when the problem is formulated using a dense belief representation, a general purpose POMDP solver is not a practical choice (Binney et al. 2013).

The sampling-based motion planning algorithms (Horsch et al. 1994; Kavraki et al. 1996; LaValle and Kuffner 2001; Bry and Roy 2011; Karaman and Frazzoli 2011; Lan and Schwager 2013) have proven successful applications in robotics. An interesting case is where the environment is fully observable (known map) and the objective is to make sequential decisions under motion and measurement uncertainty. This problem is also known as *active localization*; for a recent technique to solve such a problem see Agha-mohammadi et al. (2014, and references therein). A closely related term that is used in the literature is *belief space planning* (Kurniawati et al. 2008; Huynh and Roy 2009; Prentice and Roy 2009; Platt Jr et al. 2010; Kurniawati et al. 2011; Van Den Berg et al. 2011; Bry and Roy 2011; Valencia et al. 2013). In this series of works, the assumption of a known map can be relaxed and the environment is often represented using a set of features. The objective is to find a trajectory that minimizes the state uncertainty (the total cost) with respect to a fixed or variable (and bounded) planning horizon that can be set according to the *budget* (Indelman et al. 2015).

However, in the problem we study in this thesis, the state variables consist of the robot trajectory and a dense map of the environment. Another approach to study the problem of robotic navigation under uncertainty, is known as informative motion planning or robotic information gathering (Singh et al. 2009; Levine 2010; Binney and Sukhatme 2012; Binney et al. 2013; Hollinger and Sukhatme 2013, 2014). Rapidly-exploring information gathering (Hollinger and Sukhatme 2014), is a technique that solves the problem of informative motion planning using incremental

sampling from the workspace and by *partial ordering* of nodes builds a graph that contains informative trajectories. In this thesis, we develop this technique by considering the robot pose and the map being both partially observable. We also develop an information-theoretic criterion for the convergence of the search which allows us to perform online robotic exploration experiments in an unknown environment. The Rapidly-exploring Adaptive Search and Classification (ReASC) (Hollinger 2015) also improves on the Rapidly-exploring information gathering by allowing for real-time optimization of search and classification objective functions. However, ReASC considers discrete target locations and, similar to Platt Jr et al. (2010), resorts to the maximum likelihood assumption for future observations.

In particular, the main features of this work that differentiate the present approach from the aforementioned literature can be summarized as follows.

- We allow for dense belief representations. Therefore, we incorporate full state uncertainty, i.e. the robot pose and the map, into the planning process. As a result, the robot behavior has a strong correlation with its perception uncertainty.
- We take into account all possible future observations and do not resort to maximum likelihood assumptions. Therefore, in expectation, the randomness of future observations is addressed.
- We take into account both cost and information gain in which are included a measure of distance, motion uncertainty, and the sensing information gain. Therefore, the planning algorithm runs with respect to available sensing resources, and acting limitations.
- We interpret an information-theoretic notion of planning horizon which brings more flexibility for managing the convergence of the algorithm. This novel notion makes the planner an infinite-horizon planning technique and provides a general framework for robotic exploration.

2.3 Gaussian Processes Mapping and Exploration

The occupancy grid maps are the standard way of environment representation in robotics and are the natural choice for online implementations (Moravec and Elfes 1985; Elfes 1987; Thrun

2003b). However, they simplify the mapping problem to a set of independent cells and estimate the probability of occupancy for each cell by ignoring its correlation with neighboring cells. The map posterior is then approximated as the product of its marginals.

Kernel methods in the form of GPs framework (Rasmussen and Williams 2006) are non-parametric regression and classification techniques that have been used to model spatial phenomena (Lang et al. 2007; Vasudevan et al. 2009; Hadsell et al. 2010). Gaussian Processes have proven particularly powerful to represent the affinity of spatially correlated data, hence overcoming the traditional assumption of independence between cells, characteristic of the occupancy grid method for mapping environments (O'Callaghan et al. 2009; T O'Callaghan and Ramos 2012). The GP associated variance surface equates to a continuous representation of uncertainty in the environment, which can be used to highlight unexplored regions and optimize a robot's search plan. The continuity property of the GP map can improve the flexibility of the planner by inferring directly on collected sensor data without being limited by the resolution of the grid cell (Gan et al. 2009; Yang et al. 2013).

Gaussian Processes Occupancy Map (GPOM), in its original formulation (O'Callaghan et al. 2009; T O'Callaghan and Ramos 2012), is a batch mapping technique and the cubic time complexity of GPs is prohibitive for scenarios such as robotic navigation where a dense representation is preferred. The GPOM using a mixture of GPs is studied in Kim and Kim (2012) where the training set is divided into a number of clusters and the corresponding local occupancy maps are merged using the Bayesian Committee Machine (BCM) technique (Tresp 2000). In Kim and Kim (2013a), in order to improve the scalability of GPOMs, a divide and conquer strategy is used to partition data into clusters with manageable sizes. It also proposes to generate overlapping clusters to mitigate the discontinuity problem. In Ramos and Ott (2015), the Hilbert maps technique is proposed that is more scalable and can be updated in linear time. However, it is an approximation for continuous occupancy mapping and produces maps with less accuracy than GPOMs. The approximate methods for uncertainty propagation into GPs models through kernel functions is proposed in Girard (2004), and developed for GPOMs in O'Callaghan et al. (2010).

In the present work, we exploit GPs to develop tractable online robotic mapping and exploration techniques. We start from the problem of occupancy mapping and expand the method to exploration using geometric frontiers, and mutual information-based exploration. We propose an

approach to detect frontiers from a continuous representation of the environment that does not suffer from the issues associated with grid maps. The solution is distinctly flexible in being able to deal with sparse measurements and predicated on inferring a map posterior using Bayesian updates with a forward sensor model. We also develop uncertainty propagation into GPs models through kernel functions as well as the proposed expected sub-map technique for approximate uncertainty propagation, then incorporate it into our incremental continuous occupancy mapping framework. Finally, we integrate the developed GP-based techniques into the proposed sampling-based incremental information gathering algorithm which can provide a non-myopic solution to robotic navigation problems by considering the full state uncertainty.

This work on high-dimensional occupancy mapping and its applications can be distinguished from the literature mentioned earlier as follows.

- We develop the GPOM in an incremental form and use it for robotic exploration tasks.
- We develop the notion of probabilistic continuous frontier map extracted from the GPOM.
- We take into account both uncertain inputs and non-additive perception noise in the GP framework.
- We exploit the GP associated variance surface for fast mutual information calculations and non-myopic information-driven robotic exploration tasks.

2.4 Chapter Summary

In this chapter, a brief literature review of the current state of autonomous robotic navigation in the scope of this thesis is provided. Autonomous robots require very considerable efforts to be able to adaptively operate in real environments, and there are a large number of research works that are beyond the scope of this thesis. Here, we focus on studying the problem of robotic mapping and exploration using Gaussian processes and sampling-based information gathering algorithms. The motivations include high-dimensional map inference to accommodate structural dependencies in the environment and handling the uncertainty more accurately. In the next chapter, some background theories and techniques related to this work are discussed. Mathematical notation that is used throughout this thesis is also established.

Chapter 3

Background Theory and Techniques

We establish the mathematical notation used throughout this thesis as well as the required background theory and techniques in this chapter.

3.1 Mathematical Notation

In the present thesis probabilities and probability densities are not distinguished in general. Matrices are capitalized in bold, such as in $\boldsymbol{X}$, and vectors are in lower case bold type, such as in $\boldsymbol{x}$. Vectors are column-wise and $1:n$ means integers from 1 to n. The Euclidean norm is shown by $\|\cdot\|$. $\mathrm{tr}(\boldsymbol{X})$ and $|\boldsymbol{X}|$ denote the trace and the determinant of matrix $\boldsymbol{X}$, respectively. With a slight abuse of notion, for the sake of compactness, random variables, such as X, and their realizations, x, are sometimes denoted interchangeably where it is evident from context. $x^{[i]}$ denotes a reference to the i-th element of the variable. An alphabet such as $\mathcal{X}$ denotes a set. A subscript asterisk, such as in $\boldsymbol{x}_*$, indicates a reference to a test set quantity. The n-by-n identity matrix is denoted by $\boldsymbol{I}_n$. The sign function, and the absolute value are denoted by $\mathrm{sgn}(x)$, and $|x|$, respectively. Finally, $\mathbb{E}[\cdot]$, $\mathbb{V}[\cdot]$, and $\mathbb{Cov}[\cdot]$ denote the expected value, variance, and covariance (for random vectors) of a random variable, respectively.

3.2 Mathematical Preliminary

3.2.1 Probability Theory

Let X be a random variable that maps the sample space Ω (set of all possible outcomes) to the state space $\mathcal{X}$. Let $p(X = x) \geq 0$ be the probability of the random variable X taking a specific value x. If X is a discrete random variable, then

$$\sum_{x \in \mathcal{X}} p(X = x) = 1 \tag{3.1}$$

and for the continuous form we can write

$$\int_{x \in \mathcal{X}} p(X = x) dx = 1 \tag{3.2}$$

For the sake of simplicity, it is common to use $p(x)$ instead of $p(X = x)$ and sometimes refer to x as the random variable itself. Let Y be another random variable, the joint distribution of X and Y is $p(x, y) = p(X = x \text{ and } Y = y)$ and if X and Y are independent

$$p(x, y) = p(x)p(y) \tag{3.3}$$

The conditional probability of X given another random variable Y is defined as

$$p(x|y) = \frac{p(x, y)}{p(y)} \qquad p(y) > 0 \tag{3.4}$$

Given the joint distribution of X and Y, the marginalization rule states that the marginal distribution of X can be computed by summing (integration) over Y. The law of total probability is its variant which uses the conditional probability definition and can be written as

$$p(x) = \sum_{y \in \mathcal{Y}} p(x, y) = \sum_{y \in \mathcal{Y}} p(x|y)p(y) \tag{3.5}$$

and for continuous random variables is

$$p(x) = \int_{y \in \mathcal{Y}} p(x, y) dy = \int_{y \in \mathcal{Y}} p(x|y)p(y) dy \tag{3.6}$$

Given three random variables X, Y, and Z, Bayes rule relates the prior probability distribution, $p(x|z)$, and the likelihood function, $p(y|x,z)$, as follows.

$$p(x|y,z) = \frac{p(y|x,z)p(x|z)}{p(y|z)} \tag{3.7}$$

The term $p(x|y,z)$ is called the posterior probability distribution over X. The expected value of the random variable X is

$$\mathbb{E}[X] = \int_{\Omega} X dp = \int_{\mathcal{X}} x p(x) dx \tag{3.8}$$

and if X is discrete

$$\mathbb{E}[X] = \sum_{\mathcal{X}} x p(x) \tag{3.9}$$

Lemma 3.1 (Marginalization property of normal distribution (Von Mises 1964))**.** *Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be jointly Gaussian random vectors*

$$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{bmatrix}, \begin{bmatrix} \boldsymbol{A} & \boldsymbol{C} \\ \boldsymbol{C}^T & \boldsymbol{B} \end{bmatrix}) \tag{3.10}$$

then the marginal distribution of $\boldsymbol{x}$ is

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{A}) \tag{3.11}$$

3.2.2 Information Theory

Entropy is a measure of the uncertainty of a random variable (Cover and Thomas 1991). The entropy $H(X)$ of a discrete random variable X is defined as

$$H(X) = \mathbb{E}_{p(x)}[\log \frac{1}{p(x)}] = -\sum_{\mathcal{X}} p(x) \log p(x) \tag{3.12}$$

The joint entropy $H(X,Y)$ of discrete random variables X and Y with a joint distribution $p(x,y)$ is defined as

$$H(X,Y) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x,y) \log p(x,y) \tag{3.13}$$

The chain rule implies that

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y) \tag{3.14}$$

where $H(X|Y)$ is the conditional entropy and is defined as

$$H(Y|X) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y|x) \tag{3.15}$$

Theorem 3.2 (Chain rule for entropy)**.** *Let $X_1, X_2, ..., X_n$ be drawn according to the joint probability distribution $p(x_1, x_2, ..., x_n)$. Then*

$$H(X_1, X_2, ..., X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1}, ..., X_1) \tag{3.16}$$

Proof. Please refer to Cover and Thomas (1991) for the proof. □

The relative entropy or Kullback-Leibler Divergence (KLD) is a measure of distance between two distributions $p(x)$ and $q(x)$. It is defined as

$$D(p||q) = \mathbb{E}_{p(x)}[\log \frac{p(x)}{q(x)}] \tag{3.17}$$

Theorem 3.3 (Information inequality)**.** *Let X be a discrete random variable. Let $p(x)$ and $q(x)$ be two probability mass functions. Then*

$$D(p||q) \geq 0 \tag{3.18}$$

with equality if and only if $p(x) = q(x) \forall\ x$.

Proof. Please refer to Cover and Thomas (1991) for the proof. □

The mutual information $I(X;Y)$ is the reduction in the uncertainty of one random variable due to the knowledge of the other. The mutual information is non-negative and can be written as

$$I(X;Y) = D(p(x,y)||p(x)p(y)) = \mathbb{E}_{p(x,y)}[\log \frac{p(x,y)}{p(x)p(y)}] \tag{3.19}$$

$$I(X;Y) = H(X) - H(X|Y) \tag{3.20}$$

Corollary 3.4 (Nonnegativity of mutual information)**.** *For any two random variables X and Y,*

$$I(X;Y) \geq 0 \tag{3.21}$$

with equality if and only if X and Y are independent.

Proof. $I(X;Y) = D(p(x,y)||p(x)p(y)) \geq 0$, with equality if and only if $p(x,y) = p(x)p(y)$. □

Some immediate consequences of the provided definitions are as follows.

Lemma 3.5. *For any discrete random variable X, we have $H(X) \geq 0$.*

Proof. $0 \leq p(x) \leq 1$ implies that $\log \frac{1}{p(x)} \geq 0$. □

Theorem 3.6 (Conditioning reduces entropy)**.** *For any two random variables X and Y,*

$$H(X|Y) \leq H(X) \tag{3.22}$$

with equality if and only if X and Y are independent.

Proof. $0 \leq I(X;Y) = H(X) - H(X|Y)$. □

We now define the equivalent of the functions mentioned above for probability density functions.

Definition 3.7 (Differential entropy)**.** *Let X be a continuous random variable whose support set is $\mathcal{S}$. Let $p(x)$ be the probability density function for X. The differential entropy $h(X)$ of X is defined as*

$$h(X) = -\int_{\mathcal{S}} p(x) \log p(x) dx \tag{3.23}$$

Remark 3.8. *The differential entropy of a set of continuous random variables that have a joint distribution is also defined using Equation* (3.23)*.*

Definition 3.9 (Conditional differential entropy)**.** *Let X and Y be continuous random variables that have a joint probability density function $p(x,y)$. The conditional differential entropy $h(X|Y)$ is defined as*

$$h(X|Y) = -\int p(x,y) \log p(x|y) dx dy \tag{3.24}$$

Definition 3.10 (Relative entropy (KLD))**.** *The relative entropy (KLD) between two probability density functions p and q is defined as*

$$D(p\|q) = \int p \log \frac{p}{q} \tag{3.25}$$

Definition 3.11 (Mutual information)**.** *The mutual information $I(X;Y)$ between two continuous random variables X and Y with joint probability density function $p(x,y)$ is defined as*

$$I(X;Y) = \int p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy \tag{3.26}$$

3.2.3 Numerical Integration

Gauss-Hermite quadrature is a method for numerically approximating the integrals of the kind $\int_{-\infty}^{\infty} \exp(-x^2) f(x) dx$, as (Davis and Rabinowitz 1984; Press et al. 1996)

$$\int_{-\infty}^{\infty} \exp(-x^2) f(x) dx \approx \sum_{j=1}^{n} w^{[j]} f(x^{[j]}) \tag{3.27}$$

$$w^{[j]} = \frac{2}{[H_n'(x^{[j]})]^2} \tag{3.28}$$

$$H_{-1} = 0, \quad H_0 = \frac{1}{\pi^{1/4}}, \quad H_{j+1} = x\sqrt{\frac{2}{j+1}} H_j - \sqrt{\frac{j}{j+1}} H_{j-1} \tag{3.29}$$

$$H'_j = \sqrt{2j} H_{j-1} \tag{3.30}$$

where $x^{[j]}$ are the roots of the Hermite polynomial H_n.

3.2.4 Submodular Functions

A set function f is said to be *submodular* if $\forall \mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{S}$ and $\forall s \in \mathcal{S} \backslash \mathcal{B}$, then $f(\mathcal{A} \cup s) - f(\mathcal{A}) \geq f(\mathcal{B} \cup s) - f(\mathcal{B})$. Intuitively, this can be explained as: by adding observations to a smaller set, we gain more information. The function f has diminishing return. It is *normalized* if $f(\varnothing) = 0$ and it is *monotone* if $f(\mathcal{A}) \leq f(\mathcal{B})$. The mutual information is normalized, approximately monotone, and submodular (Krause et al. 2008).

3.2.5 Gaussian Processes

In this subsection, a brief introduction of Gaussian Processes is presented. The notation and concept are mainly from the book Gaussian Processes for Machine Learning by Rasmussen and Williams (2006). Supervised learning is the problem of learning input-output mappings from a training dataset. Based on the nature of the output the problem is known as regression (continuous outputs) or classification (discrete outputs). A Gaussian process is a generalization of the Gaussian probability distribution. Random variables which are scalars or vectors (for multivariate distributions) are described by a probability distribution whilst a stochastic process governs the properties of functions. The Gaussian process framework unites a sophisticated and consistent view with computational tractability. In Rasmussen and Williams (2006), from function-space point of view a GP is formally defined as: *A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.*

GPs are a non-parametric Bayesian regression technique in the sense that they do not explicitly define a functional relationship between inputs and outputs. Instead, statistical inference is employed to learn dependencies between points in a dataset. Define a training set $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)|i = 1:n\}$ which consists of a d-dimensional input vector $\boldsymbol{x}$ and a scalar output (target) value y for n observations. The joint distribution of the observed target values, $\boldsymbol{y}$, and the function values (the latent variable), $\boldsymbol{f}_*$, at the query points can be written as

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n & \boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}_*) \\ \boldsymbol{K}(\boldsymbol{X}_*,\boldsymbol{X}) & \boldsymbol{K}(\boldsymbol{X}_*,\boldsymbol{X}_*) \end{bmatrix}) \tag{3.31}$$

where $\boldsymbol{X}$ is the $d \times n$ design matrix of aggregated input vectors $\boldsymbol{x}$, $\boldsymbol{X}_*$ is a $d \times n_*$ query points matrix, $\boldsymbol{K}(\cdot,\cdot)$ is the GP covariance matrix, $\boldsymbol{I}_n$ is the n-by-n identity matrix, and σ_n^2 is the variance of the observation noise which is assumed to have an independent and identically distributed Gaussian distribution. The predictive conditional distribution for a single query point $f_*|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x}_* \sim \mathcal{N}(\mathbb{E}[f_*], \mathbb{V}[f_*])$ can be derived as

$$\mu = \mathbb{E}[f_*] = \boldsymbol{k}(\boldsymbol{X},\boldsymbol{x}_*)^T [\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n]^{-1} \boldsymbol{y} \tag{3.32}$$

$$\sigma = \mathbb{V}[f_*] = k(\boldsymbol{x}_*,\boldsymbol{x}_*) - \boldsymbol{k}(\boldsymbol{X},\boldsymbol{x}_*)^T [\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n]^{-1} \boldsymbol{k}(\boldsymbol{X},\boldsymbol{x}_*) \tag{3.33}$$

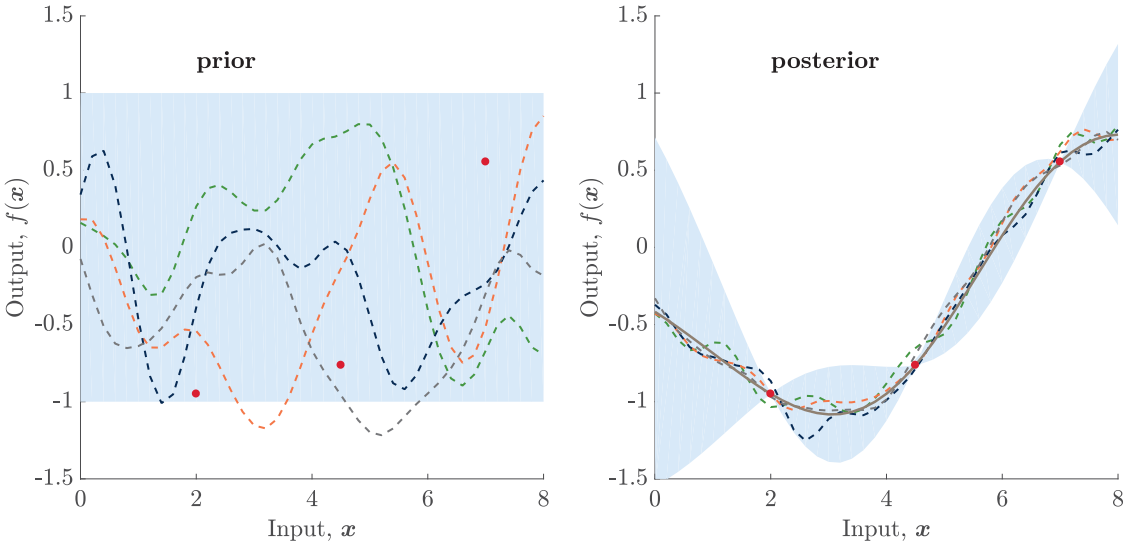


FIGURE 3.1: An illustrative example of the Gaussian Process prior and predictive posterior over functions. In the left figure, four functions are randomly sampled from the GP prior where three observations points are shown in red. In the right figure, four functions are drawn at random from the GP predictive posterior. The uncertainty of the predictive distribution at the training points is lower, whereas GP predicts the underlying process at other regions with higher uncertainties. The predictive mean is also shown in the right figure using a gray solid line. The light blue areas correspond to 95% confidence regions.

An illustrative example of the GP is shown in Figure 3.1. The figure shows randomly drawn functions from the GP prior and posterior.

There are some free parameters in a covariance function. In general, these parameters are known as *hyperparameters*. The model selection problem refers to the problem of setting of hyperparameters within a family, and comparing across different families. Selection of a covariance function and its parameters is called training of a Gaussian process. These problems and various methods to determine the hyperparameters from training data are discussed in detail in Rasmussen and Williams (2006), chapter 5. The hyperparameters of the covariance and mean function, $\boldsymbol{\theta}$, can be computed by minimization of the Negative Log of the Marginal Likelihood (NLML) function

$$\log p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2\boldsymbol{I}_n)^{-1}\boldsymbol{y} - \frac{1}{2}\log|K(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2\boldsymbol{I}_n| - \frac{n}{2}\log 2\pi \tag{3.34}$$

For current application in this thesis, optimization of the hyperparameters can be undertaken at each increment, but in practice the initial optimization of the hyperparameters produces a set

of learned parameters than can be used in subsequent increments. This approach is particularly suitable for real-time scenarios.

3.2.5.1 Covariance Function

Covariance functions are the main part of any GPs. We define a covariance function using the kernel definition as following.

Definition 3.12 (Covariance function)**.** *Let $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{x}' \in \mathcal{X}$ be a pair of inputs for a function $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ known as kernel. A kernel is called a covariance function, as the case in Gaussian processes, if it is symmetric, $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}', \boldsymbol{x})$, and Positive Semi-Definite (PSD):*

$$\int k(\boldsymbol{x}, \boldsymbol{x}') f(\boldsymbol{x}) f(\boldsymbol{x}') d\mu(\boldsymbol{x}) d\mu(\boldsymbol{x}') \geq 0 \tag{3.35}$$

for all $f \in L_2(\mathcal{X}, \mu)$.

Given a set of input points $\{\boldsymbol{x}|i = 1 : n\}$, a covariance matrix can be constructed using $\boldsymbol{K}^{[i,j]} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ as its entries.

3.2.5.2 Useful Kernels

The Squared Exponential (SE) covariance function has the form

$$k(r) = \exp(-\frac{r^2}{2l^2}) \tag{3.36}$$

where $r = ||\boldsymbol{x} - \boldsymbol{x}_*||$ is the distance between two input arguments of the covariance function and l is the *characteristic length-scale*. This covariance function is the most common kernel used in GPs and is infinitely differentiable.

The Matérn family of covariance functions (Stein 1999) has proven powerful features to model structural correlations (Ghaffari Jadidi et al. 2013a; Kim and Kim 2013b; Ghaffari Jadidi et al. 2014; Kim and Kim 2015). For a single query point $\boldsymbol{x}_*$ the function is given by

$$k(r) = \frac{1}{\Gamma(\nu) 2^{\nu-1}} \left[\frac{\sqrt{2\nu} r}{l} \right]^{\nu} K_{\nu} \left(\frac{\sqrt{2\nu} r}{l} \right) \tag{3.37}$$

where $\Gamma(\cdot)$ is the Gamma function, $K_\nu(\cdot)$ is the modified Bessel function of the second kind of order ν, l is the characteristic length scale, and ν is a positive parameter used to control the smoothness of the covariance. In the limit for $\nu \to \infty$ this covariance function converges to the SE kernel. The widely-used cases for machine learning are $\nu = 3/2$ and $\nu = 5/2$ and the function becomes of the form

$$k_{\nu=3/2}(r) = (1 + \frac{\sqrt{3}r}{l})\exp(-\frac{\sqrt{3}r}{l}) \tag{3.38}$$

$$k_{\nu=5/2}(r) = (1 + \frac{\sqrt{5}r}{l} + \frac{5r^2}{3l^2})\exp(-\frac{\sqrt{5}r}{l}), \tag{3.39}$$

for the case of $\nu = 1/2$ the function behaves very rough (Rasmussen and Williams 2006) and is known as the exponential covariance function.

An intrinsically sparse covariance function is developed in Melkumyan and Ramos (2009). The function is a *stationary* covariance function and its sparseness is controlled through learning the hyperparameters. The extension of the function to multiple dimensions using Mahalanobis distance is

$$k_{Sparse}(r) = \begin{cases} \sigma_0[\frac{2+\cos(2\pi r)}{3}(1-r) + \frac{1}{2\pi}\sin(2\pi r)] & \text{if} \quad r < 1 \\ 0 & \text{if} \quad r \geq 1 \end{cases} \tag{3.40}$$

where in this case $r = \sqrt{(\boldsymbol{x} - \boldsymbol{x}_*)^T \boldsymbol{L}(\boldsymbol{x} - \boldsymbol{x}_*)}$, $\sigma_0 > 0$ and $\boldsymbol{L}$ is a positive semi-definite matrix that can be set as the diagonal matrix of the inverse square of the characteristic length-scales.

Examples of the SE, Matérn ($\nu = 5/2$), and Sparse covariance functions as the distance parameter r increases are shown in Figure 3.2. The functions are also plotted in the kernel space in Figure 3.3. For all examples, a unit length-scale is used. The diminishing behavior of the Sparse covariance function as the distance parameter exceeds 1 can be seen in the plot which in the corresponding kernel space is represented by mainly non-zero elements on the diagonal part of the covariance matrix. In general, a smooth kernel such as the SE is suitable to model highly correlated training data, while less smooth kernels such as Matérn and Sparse can be used for the opposite purpose.

3.2.5.3 GP libraries

The open source GP library in (Rasmussen and Williams 2006) is developed in MATLAB and provides a broad range of options for mean, covariance, likelihood, and inference functions. There

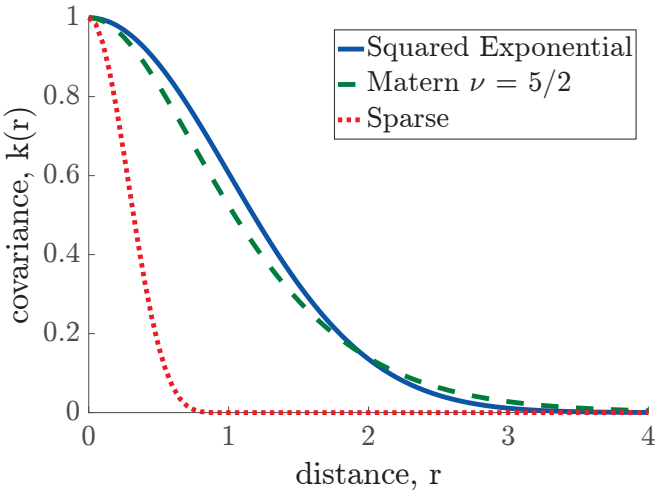
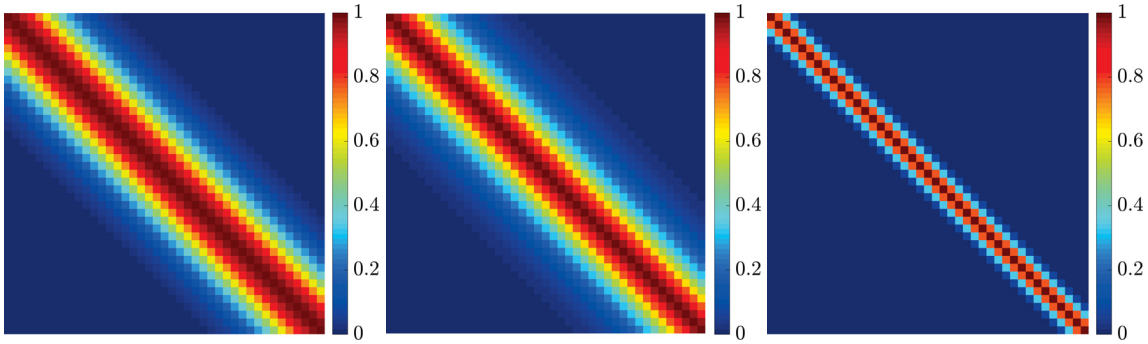


FIGURE 3.2: Illustrative examples of the SE, Matérn ($\nu = 5/2$), and Sparse covariance function as the distance parameter is increased from 0 to 4. The length-scale parameter is set to one.



(A) SE (B) Matérn $\nu = 5/2$ (C) Sparse

FIGURE 3.3: Corresponding function values from Figure 3.2 in the kernel space.

are also different approximate inference methods available. The open source library pyGPs (Neumann et al. 2014) is developed in Python and can be used independently or for online experiments using Robot Operating System (ROS) (Quigley et al. 2009).

3.2.6 System Dynamics

The equation of motion of the robot is governed by the nonlinear partially observable equation as follows.

$$\boldsymbol{x}_{t+1}^{-} = f(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{w}_t) \quad \boldsymbol{w}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_t) \tag{3.41}$$

moreover, with appropriate linearization at the current state estimate, we can predict the state covariance matrix as

$$\Sigma_{t+1}^{-} = \boldsymbol{F}_t \Sigma_t \boldsymbol{F}_t^T + \boldsymbol{W}_t \boldsymbol{Q}_t \boldsymbol{W}_t^T \tag{3.42}$$

where $\boldsymbol{F}_t = \frac{\partial f}{\partial \boldsymbol{x}}|_{\boldsymbol{x}_t, \boldsymbol{u}_t}$ and $\boldsymbol{W}_t = \frac{\partial f}{\partial \boldsymbol{w}}|_{\boldsymbol{x}_t, \boldsymbol{u}_t}$ are the Jacobian matrices calculated with respect to $\boldsymbol{x}$ and $\boldsymbol{w}$, respectively.

3.3 Basic Techniques

In this section, we briefly explain two major methods that are related to the work in this thesis. First we introduce the occupancy grid mapping method which is used for traditional mapping and exploration techniques in order to make comparison. Next, the Pose SLAM algorithm is explained. We use Pose SLAM throughout the experiments to provide localization data for mapping and exploration. It is important to solve the SLAM problem initially and observe the effect of each mapping and exploration technique on the consistency of the SLAM algorithm and localization accuracy. In Figure 3.4, the examples of Pose SLAM graph and OGM are shown.

3.3.1 Occupancy Grid Maps

The objective of the occupancy grid mapping is to estimate the posterior over maps given the data $p(m|z_{1:t}, \boldsymbol{x}_{1:t})$ in which $z_{1:t}$ is the set of all measurements up to time t and $\boldsymbol{x}_{1:t}$ is the robot trajectory. At any location, the map cell $m^{[i]}$ is considered as a binary random variable as it can represent an empty ($p = 0$) or occupied cell ($p = 1$). The fundamental assumption in the OGM algorithm is the independence between cells (Moravec and Elfes 1985; Elfes 1987). Therefore, the map posterior can be expressed as the product of the individual marginal probabilities of the map cells

$$p(m|z_{1:t}, \boldsymbol{x}_{1:t}) = \prod_i p(m^{[i]}|z_{1:t}, \boldsymbol{x}_{1:t}) \tag{3.43}$$

In order to avoid numerical instability for probabilities near zero or one, it is common to use *log odds* representation. Using the *binary Bayes filter* (Thrun et al. 2005) to estimate each grid cell, we can write

$$l_0^{[i]} = \log \frac{p(m^{[i]})}{1 - p(m^{[i]})} \tag{3.44}$$

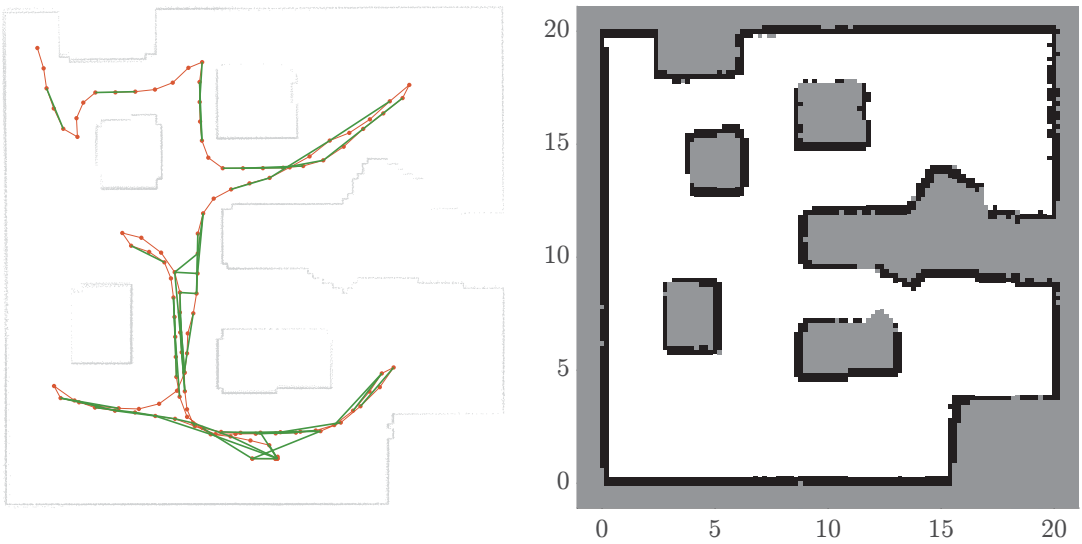


FIGURE 3.4: An illustrative example of the Pose SLAM graph (left), and its associated dense occupancy grid map (right). In the Pose SLAM graph, the robot poses (vertices) together with their corresponding odometric measurements links are shown in red. The green links (edges) show loop-closures computed from sensor registrations. In the occupancy grid map, black, gray, and white colors indicate occupied, unknown, and free cells, respectively. Map dimensions are in meters.

$$l_{inv}^{[i]} = \log \frac{p(m^{[i]}|z_t, x_t)}{1 - p(m^{[i]}|z_t, x_t)} \tag{3.45}$$

$$l_t^{[i]} = \log \frac{p(m^{[i]}|z_{1:t}, \boldsymbol{x}_{1:t})}{1 - p(m^{[i]}|z_{1:t}, \boldsymbol{x}_{1:t})} \tag{3.46}$$

$$l_t^{[i]} = l_{t-1}^{[i]} + l_{inv}^{[i]} - l_0^{[i]} \tag{3.47}$$

where $l_0^{[i]}$ is calculated from map prior probability $p(m^{[i]})$, $l_{inv}^{[i]}$ is known as the inverse sensor model in which it assigns an occupancy value to each cell that is in the perception field of measurement z_t.

3.3.2 Pose SLAM

Pose SLAM (Ila et al. 2010) is a pose graph SLAM algorithm that generates a graph in which the nodes represents robot poses and the edges are relative measurements from odometry or sensor registration. It is an information-based approach that only considers highly informative loop-closures and non-redundant poses. The objective is to estimate the robot trajectory with

a canonical parametrization $p(\boldsymbol{x}_{1:t}) = \mathcal{N}^{-1}(\boldsymbol{\eta}, \boldsymbol{\Lambda})$ in order to maintain the sparsity of the graph. The information vector is related to the mean vector $\boldsymbol{\eta} = \boldsymbol{\Lambda}\boldsymbol{\mu}$ and the information matrix is the inverse of the covariance matrix $\boldsymbol{\Lambda} = \Sigma^{-1}$.

In the online Pose SLAM (Eustice et al. 2006; Ila et al. 2007), the state transition step connects a new pose to the immediate previous pose using an odometry measurement u_t and Bayesian factorization. In the state update step, using sensor registration z_t, the algorithm tries to find a link between the current pose and any other pose. Each state update changes the entire state estimate and the mean vector is required to be recovered in order to compute the Jacobian matrices for the next filtering step. Therefore, the algorithms consider the mutual information gain of a candidate edge before adding it to the graph. In this way, the size of the state can be reduced.

3.4 Chapter Summary

In this chapter, we established the mathematical notation as well as some preliminary and basic definitions and techniques that will be used throughout this thesis. In the next chapter, we develop Gaussian process occupancy mapping techniques for both mapping and exploration purposes.

Chapter 4

Gaussian Processes Continuous Occupancy Mapping

Autonomous mobile robots are required to generate a spatial representation of the robot environment, known as the mapping problem. Solving this problem is an integral part of all autonomous navigation systems as it encapsulates the knowledge of the robot about its surrounding. In robotic navigation tasks, a map that indicates occupied areas of the environment is required. Furthermore, it is desirable that such maps be generated autonomously where the robot explores new regions of an unknown environment and at the same time maximizes map accuracy. This is known as the autonomous exploration problem in robotics.

The objective is to estimate the occupancy status of any desired locations in the environment. However, unlike occupancy grid maps that ignore correlation present in data, we use GPs as a high-dimensional regression technique to accommodate this matter. GPs have cubic time complexity due to the inversion of the covariance matrix of the size of training data which is extremely prohibitive. As a result, to map an environment faster than batch mapping with any size, i.e. as far as the available memory allows, a systematic framework is needed to incorporate information incrementally into the map while the accuracy remains acceptable. In this chapter, the continuous occupancy mapping technique which exploits Gaussian processes at its core is developed. We start from basic GP occupancy mapping and discuss the challenges and limitations of using the

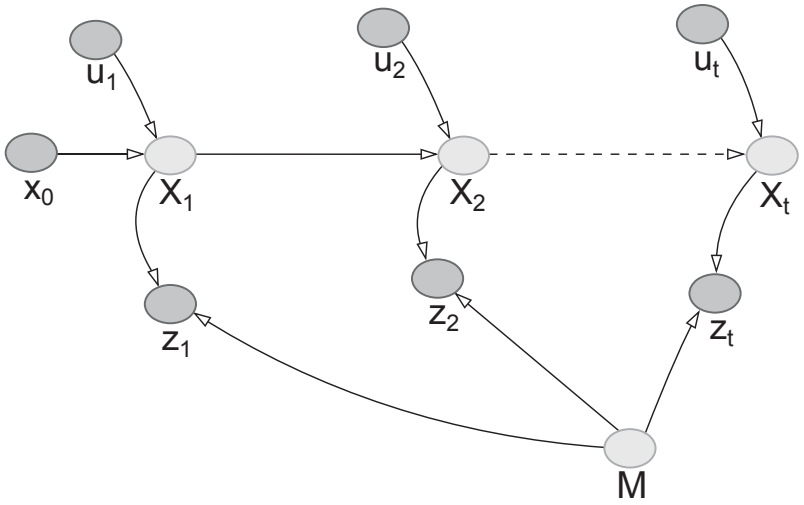


FIGURE 4.1: Dynamic Bayes network of the SLAM process. Given initial pose $\boldsymbol{x}_0$, control actions $\boldsymbol{u}_{1:t}$ and observations $\boldsymbol{z}_{1:t}$, Pose SLAM estimates the robot trajectory $\boldsymbol{x}_{1:t}$. Therefore, in the occupancy mapping process robot poses are known (the uncertainty is ignored) and the map and measurements are considered as random variables. Shaded nodes are observed.

map in practice. The method is developed from the autonomous robotic navigation perspective in both batch and incremental form with quantitative performance comparison.

Remark 4.1. *Since we eventually devise a map building technique based on two GPs to produce a map suitable for exploration, throughout this chapter we define separate variables and sets for occupied and free areas of the map.*

We are concerned with autonomous exploration when the location of the robot is available through an appropriate strategy such as Pose SLAM Ila et al. (2010). Therefore, we first consider the map inference using the current estimate of the robot pose. Later, we discuss possible extensions of the problem of uncertainty propagation into the occupancy mapping process.

Figure 4.1 shows the graphical model of the SLAM process. Given initial pose, control actions, and relative range and bearing observations, initially, we solve the robot localization problem. Afterward, the map inference is done using known robot poses and corresponding observations. Before formal statement of the problem, we clarify the following assumptions.

Assumption 4.2 (Static occupancy map representation)**.** *The environment that the robot navigates in is static.*

Assumption 4.3 (Gaussian occupancy map points)**.** *Any sampled point from the occupancy map representation of the environment is a random variable whose distribution is Gaussian.*

4.1 Problem Statement and Formulation

Let $\mathcal{M}$ be the set of possible occupancy maps. We consider the map of the environment to be static and as an n_m-tuple random variable $(M^{[1]}, ..., M^{[n_m]})$ whose elements are described by a normal distribution $m^{[i]} \sim \mathcal{N}(\mu^{[i]}, \sigma^{[i]})$, $i \in \{1 : n_m\}$. Let $\mathcal{Z} \subset \mathbb{R}_{\geq 0}$ be the set of possible range measurements. The observation consists of an n_z-tuple random variable $(Z^{[1]}, ..., Z^{[n_z]})$ whose elements can take values $\boldsymbol{z}^{[k]} \in \mathcal{Z}$, $k \in \{1 : n_z\}$. Let $\mathcal{X} \subset \mathbb{R}^2$ be the set of spatial coordinates to build a map on. Let $\boldsymbol{x}_o^{[k]} \in \mathcal{X}_o \subset \mathcal{X}$ be an observed occupied point by the k-th sensor beam from the environment which can be calculated by transforming the local observation $\boldsymbol{z}^{[k]}$ to the global frame using the robot pose $\boldsymbol{x}_t \in \mathrm{SE}(2)$. Let $\boldsymbol{X}_f^{[k]} \in \mathcal{X}_f \subset \mathcal{X}$ be the matrix of sampled unoccupied points from a line segment with the robot pose and corresponding observed occupied point as its endpoints. Let $\mathcal{D} = \mathcal{D}_o \cup \mathcal{D}_f$ be the set of all training points. We define a training set of occupied points $\mathcal{D}_o = \{(\boldsymbol{x}_o^{[i]}, y_o^{[i]}) \mid i = 1 : n_o\}$ and a training set of unoccupied points $\mathcal{D}_f = \{(\boldsymbol{x}_f^{[i]}, y_f^{[i]}) \mid i = 1 : n_f\}$ in which $\boldsymbol{y}_o = [y_o^{[i]}, ..., y_o^{[n_o]}]^T$ and $\boldsymbol{y}_f = [y_f^{[i]}, ..., y_f^{[n_f]}]^T$ are target vectors and each of their elements can belong to the set $\mathcal{Y} = \{-1, +1\}$ where -1 and $+1$ corresponds to unoccupied and occupied locations, respectively, n_o is the total number of occupied points, and n_f is the total number of unoccupied points. Given the robot pose $\boldsymbol{x}_t$ and observations $Z_t = \boldsymbol{z}_t$, we wish to estimate $p(M = m \mid \boldsymbol{x}_t, Z_t = \boldsymbol{z}_t)$. The map can be inferred as a Gaussian process by defining the process as the function $y : \mathcal{X} \to \mathcal{M}$, therefore

$$y(\boldsymbol{x}) \sim \mathcal{GP}(f_m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')) \tag{4.1}$$

It is often the case that we set the mean function $f_m(\boldsymbol{x})$ as zero, unless it is mentioned explicitly that $f_m(\boldsymbol{x}) \neq 0$. For a given query point in the map, $\boldsymbol{x}_*$, GP predicts a mean, μ, and an associated variance, σ. We can write

$$m^{[i]} = y(\boldsymbol{x}_*^{[i]}) \sim \mathcal{N}(\mu^{[i]}, \sigma^{[i]}) \tag{4.2}$$

To show a valid probabilistic representation of the map $p(m^{[i]})$, the classification step, a logistic regression classifier (Rasmussen and Williams 2006, Sections 3.1 and 3.2), (Murphy 2012, Chapter 8), squashes data into the range $[0, 1]$.

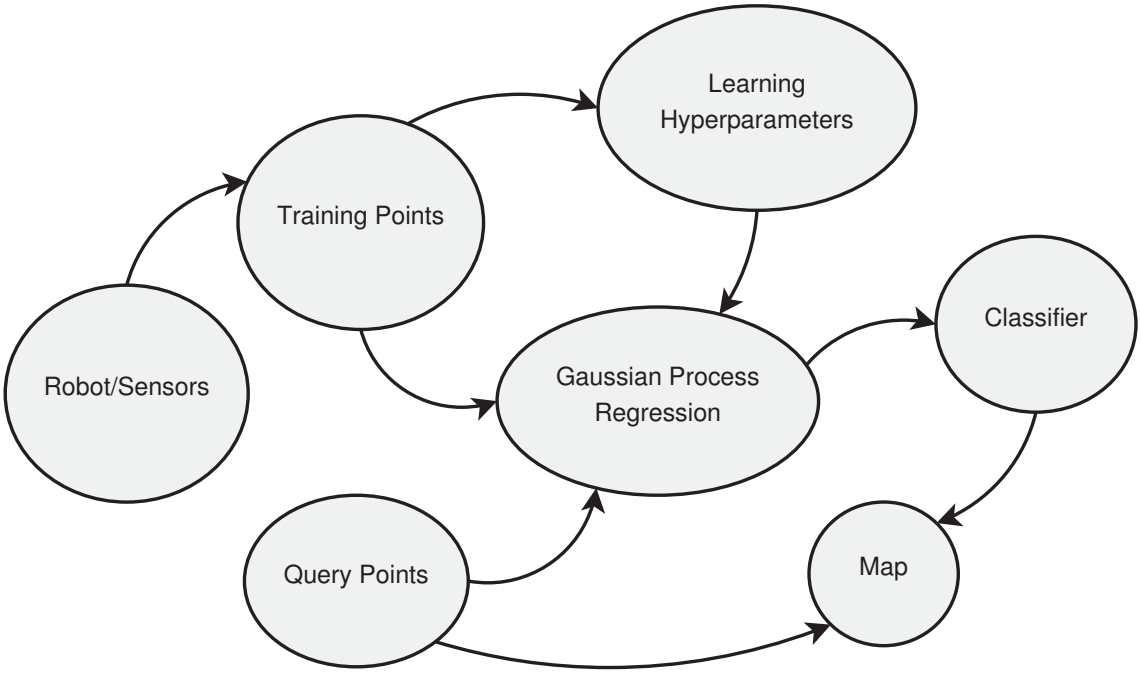


FIGURE 4.2: Sub-tasks of continuous occupancy mapping using Gaussian process regression. The classifier provides a valid probabilistic representation of the map and the map representation is based on the distribution of query data.

4.2 Gaussian Processes Occupancy Maps

We employ the continuous occupancy mapping technique which benefits from GPs to learn the correlation between map points and infer the map in its original high dimensional space. The objective is to exploit structural correlation present in the environment. We develop the mapping method for range-sensing mobile robots which are widely used in robotics. The sub-tasks of continuous occupancy mapping using GPs are depicted in Figure 4.2. GP regression is at the core of the technique for solving the COM problem. However, the nature of occupancy mapping is a binary classification problem due to the occupancy status of any points in the environment. As a result, a classifier is often used to post-process the regression results for a valid probabilistic representation. The map representation also depends on the distribution and the data structure of query points. In the following subsections, we explain the details of each sub-task.

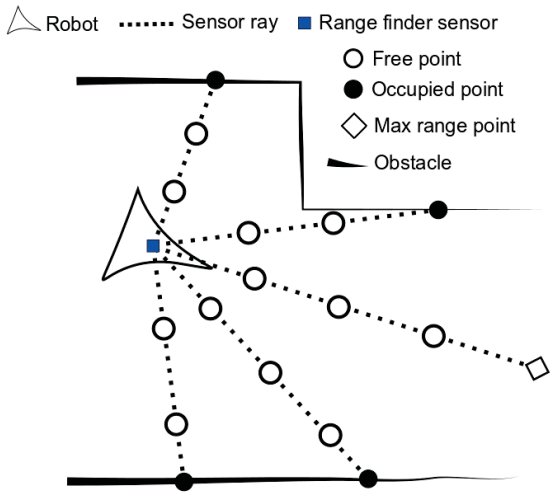


FIGURE 4.3: Conceptual illustration of the robot, the environment, and observations. Training data consists of free and occupied points labeled $y_- = -1$ and $y_+ = +1$ respectively. Free points are sampled along each beam, i.e. negative sensor information whilst occupied points are directly observable.

4.2.1 Sensor Model and Training Data

The robot is assumed to be equipped with a 2D range-finder sensor. The raw measurements include points returned from obstacle locations. For any sensor beam, the distance from the sensor position to the detected obstacle along that beam indicates a line from the unoccupied region of the environment. To build training data points for the unoccupied part of the map, it is required to sample along the aforementioned line. Figure 4.3 shows the conceptual illustration of the environment and training points generation.

A sensor beam $\boldsymbol{z}_t = (\boldsymbol{z}_t^{[1]}, ..., \boldsymbol{z}_t^{[n_z]})$ has n_z range observations at a specific bearing depending on the density of the beam. The observation model for each $\boldsymbol{z}_t^{[k]}$ can be written as

$$\boldsymbol{z}_t^{[k]} = \begin{bmatrix} r_t^{[k]} \\ \alpha_t^{[k]} \end{bmatrix} = h(\boldsymbol{x}_t, \boldsymbol{x}_o^{[k]}) + \boldsymbol{v}, \quad \boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}) \tag{4.3}$$

$$h(\boldsymbol{x}_t, \boldsymbol{x}_o^{[k]}) \triangleq \begin{bmatrix} \sqrt{(\boldsymbol{x}_o^{[k,1]} - \boldsymbol{x}_t^{[1]})^2 + (\boldsymbol{x}_o^{[k,2]} - \boldsymbol{x}_t^{[2]})^2} \\ \arctan(\boldsymbol{x}_o^{[k,2]} - \boldsymbol{x}_t^{[2]}, \boldsymbol{x}_o^{[k,1]} - \boldsymbol{x}_t^{[1]}) - \boldsymbol{x}_t^{[3]} \end{bmatrix} \tag{4.4}$$

where $r_t^{[k]}$ is the range measurement from the k-th sensor beam and $\alpha_t^{[k]}$ is the corresponding angle of $r_t^{[k]}$. The observation model noise $\boldsymbol{v}$ is assumed to be Gaussian with zero mean and

covariance $\boldsymbol{R}$. To find $\boldsymbol{x}_o^{[k]}$ which is in the map space, the inverse model can be calculated as

$$\boldsymbol{x}_o^{[k]} = \boldsymbol{x}_t^{[1:2]} + r_t^{[k]} R(\boldsymbol{x}_t^{[3]}) \begin{bmatrix} \cos(\alpha_t^{[k]}) \\ \sin(\alpha_t^{[k]}) \end{bmatrix} \tag{4.5}$$

where $R(\boldsymbol{x}_t^{[3]}) \in \mathrm{SO}(2)$ indicates a 2×2 rotation matrix.

Having defined the observed occupied points in the map space, now we can construct the training set of occupied points as $\mathcal{D}_o = \{(\boldsymbol{x}_o^{[k]}, y_o^{[k]}) \mid k = 1 : n_z\}$. One simple way to build the free area training points is to uniformly sample along the line segment, $l_z^{[k]}$, with the robot position and any occupied point $\boldsymbol{x}_o^{[k]}$ as its end points. Therefore,

$$\boldsymbol{X}_f^{[k,j]} = \boldsymbol{x}_t^{[1:2]} + \delta_j R(\boldsymbol{x}_t^{[3]}) \begin{bmatrix} \cos(\alpha_t^{[k]}) \\ \sin(\alpha_t^{[k]}) \end{bmatrix} \tag{4.6}$$

where $\delta_j \sim \mathcal{U}(0, r_t^{[k]}) \quad j = 1 : n_f^{[k]}$, $\mathcal{U}(0, r_t^{[k]})$ is a uniform distribution with the support $[0, r_t^{[k]}]$ and $n_f^{[k]}$ is the desired number of samples for the k-th sensor beam. $n_f^{[k]}$ can be a fixed value for all the beams or variable, e.g. a function of the line segment length $|l_z^{[k]}| = r_t^{[k]}$. In the case of a variable number of points for each beam, it is useful to set a minimum value $n_{f\,min}$. Therefore we can write

$$n_f^{[k]} \triangleq \max(\{n_{f\,min}, s_l(r_t^{[k]})\}) \tag{4.7}$$

where $s_l(\cdot)$ is a function that adaptively generates a number of sampled points based on the input distance. This minimum value controls the sparsity of the training set of unoccupied points. Alternatively, we can select a number of equidistant points instead of sampling. However, as the number of training points increases, the computational time grows cubicly. We can construct the training set of unoccupied points as $\mathcal{D}_f = \bigcup_{i=1}^{n_z} \mathcal{D}_f^{[i]}$ where $\mathcal{D}_f^{[i]} = \{(\boldsymbol{X}_f^{[k]}, \boldsymbol{y}_f^{[k]}) \mid k = 1 : n_z\}$ and $\boldsymbol{y}_f^{[k]} = [y_f^{[1]}, ..., y_f^{[n_f^{[k]}]}]^T$.

Remark 4.4. *Generally speaking, building maps using GPs can handle sparse sensor observations and consequently sparse training data. However, in practice, the kernel function describes the correlation between training points. A smooth kernel such as the SE can cover a larger area with fewer training points, and a rough kernel such as Matérn ($\nu = 1/2$) can only cover the vicinity of sparse*

training points. Therefore, model selection is crucial to fully exploit capabilities of GPs. Furthermore, hyperparameters have a significant effect on the shape of the map.

Since we only resort to a locally optimal solution of minimizing NLML, one can not expect suitable outputs from every run, and the solution needs to be checked carefully. We will discuss these concerns in the following subsection.

4.2.2 Model Selection and Learning Hyperparameters

Gaussian processes are non-parametric models in a sense that one can learn free parameters of mean, covariance, and likelihood functions (hyperparameters) through optimizing a cost function. Practically, this is done by marginalization of the latent function and minimization of NLML function (Equation 3.34), repeated here for convenience

$$\log\ p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2\boldsymbol{I}_n)^{-1}\boldsymbol{y} - \frac{1}{2}\log\ |K(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2\boldsymbol{I}_n| - \frac{n}{2}\log\ 2\pi$$

In Equation (3.34), the first term $-\frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2\boldsymbol{I}_n)^{-1}\boldsymbol{y}$ corresponds to data-fit, the second term $\frac{1}{2}\log\ |K(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2\boldsymbol{I}_n|$ penalizes the model complexity, and the last term is a constant.

Remark 4.5. *Note that the selection of NLML function for the optimization problem is entirely optional. However due to the marginalization property of the multivariate normal distribution, the latent variable $\boldsymbol{f}_*$ can be conveniently marginalized.*

Since we seldom have sufficient information about a process, this is a useful approach as it provides flexibility for model selection. However, the problem of minimizing NLML is an ill-posed problem and using maximum likelihood estimate we can often find a local minimum. Therefore, it is always possible that the solution suffers from over-fitting, especially if the number of hyperparameters is large (Rasmussen and Williams 2006). As we often choose the mean function to be zero, we are mainly concerned with the selection of a suitable kernel for the spatial mapping problem. The challenge here is that the structural shape of the environment, especially indoor environments, can have very diverse forms. There are often sudden variations from unoccupied to occupied areas which resemble the Heaviside step function which is considered a non-trivial pattern to be learned in machine learning (Calandra et al. 2014). Moreover, the problem becomes

more difficult when there is a vast area of free space, disconnected and thin obstacles, and connections to narrow passages. The reason is that for a large unoccupied space, a smooth kernel can model the area efficiently with a very limited number of observations; however, the same kernel fails to model walls, entrances, and small obstacles and their connections accurately.

As a result, inferring a high-quality map compatible with the actual shape of the environment can be non-trivial (see Figure 9 in T O'Callaghan and Ramos (2012) and Figure 3 in Kim and Kim (2013a)). Although considering correlations of map points through regression results in handling sparse measurements, training a unique GP for both occupied and free areas has two major challenges:

- It limits the selection of an appropriate kernel that suits both occupied and unoccupied regions of the map, effectively resulting in poorly extrapolated obstacles or low-quality free areas.
- Most importantly, it leads to a mixed variance surface. In other words, it is not possible to disambiguate between boundaries of occupied-unknown and free-unknown space, unless the continuous map is thresholded (see Figure 6 in T O'Callaghan and Ramos (2012)).

The first problem is directly related to the inferred map quality, while the second is a challenge for exploration using continuous occupancy maps. The integral kernel technique O'Callaghan and Ramos (2011) can mitigate the first deficiency mentioned earlier; however, the integration over GPs kernels is computationally demanding and results in less tractable methods. To address these problems we propose training two separate GPs, one for free areas and one for obstacles, and merge them to build a unique COM.

Conjecture 4.6. *Intuitively, the structural correlation is only related to the occupied set of points. Therefore, exactly for the reason that the problem is defined as a binary classification, correlating the unoccupied and occupied regions using one kernel decreases the accuracy of the map.*

Nevertheless, we consider both cases of mapping using a single GP, which we refer to it as Incremental Gaussian Processes Occupancy Map (I-GPOM), and two GPs, I-GPOM2. We also present and compare results to support our idea. In our GP mapping approach, optimization of hyperparameters is performed once at the beginning of each experiment by minimization of the NLML.

For the prevailing case of multiple runs in the same environment, the optimized values can then be loaded off-line.

Remark 4.7. *In heteroscedastic processes, noise is state dependent. Heteroscedastic Gaussian process regression (Goldberg et al. 1998; Kersting et al. 2007) can be an alternative to model the structural correlation more accurately. However, to model the prediction uncertainty, they require a second GP in addition to the GP governing the noise-free output value. The computational cost is roughly twice that of the standard GP (Ko and Fox 2009; Titsias and Lázaro-Gredilla 2011). To keep the proposed mapping algorithm computationally attractive, we do not consider the heteroscedasticity in training data; however, applying these techniques to the problem at hand is an interesting direction to follow.*

4.2.3 Regression and Classification

GP regression is the main step in the continuous occupancy mapping, where given the earlier discussed inputs we can learn the spatial correlation between map points and calculate the map posterior without marginalization, as opposed to OGM. By looking at Equations 3.32 and 3.33, it is evident that only mean values depend on observations (target vector $\boldsymbol{y}$). The conditional marginal variance only depends on the input; a property of GPs. In spatial mapping, it implies that spatial correlation is independent of observations. This property makes the method more general as we do not need to redevelop the basics and solely the target vector needs to be defined appropriately.

In general, there is no guarantee that predicted GPs mean values will be constrained to the target values. Following the idea in binary discriminative classification (see Sections 3.1 and 3.2 in Rasmussen and Williams (2006) and Chapter 8 in Murphy (2012)), a logistic function (response function) squashes the predicted outputs into the range $[0, 1]$ and guarantees a valid probabilistic interpretation. Therefore,

$$p(m^{[i]}|\boldsymbol{x}_*^{[i]}, w^{[i]}) = \frac{1}{1+\exp(-w^{[i]}\mu^{[i]})} \tag{4.8}$$

where $w^{[i]} \triangleq \gamma\sqrt{\lambda^{[i]}}$ denotes the required weights, $\lambda^{[i]} \triangleq \sigma_{min}/\sigma^{[i]}$ is the bounded information associated with location i, $\gamma > 0$ is a constant to control the sigmoid shape, and σ_{min} is the minimum estimated variance by the GP.

An alternative solution is discussed in T O'Callaghan and Ramos (2012) through employing a probabilistic least-square classifier (see Section 6.5 in Rasmussen and Williams (2006)). However, it requires a unique covariance matrix inversion for each point which can be a bottleneck for long-term exploration experiments.

Remark 4.8. *The actual representation of the map depends on the distribution of query points. It is often the case to use uniformly distributed points (e.g. a grid). Generally speaking, query points can have any desired distributions. However, building the map over a grid facilitates comparison with standard occupancy grid-based methods, i.e. at similar map resolutions.*

Remark 4.9. *Even though the common way for map representation is using a dense set of points with a particular distribution that is suitable for navigation tasks, there is no restriction for any other desired representation such as approximate belief representations in Charrow et al. (2014), that can be useful for other applications such as predictions.*

The main bottleneck in GP regression is computation of the term $[\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n]^{-1}$ where the covariance matrix of training data has to be inverted. This limitation reveals itself more when one tries to use a large number of training data or batch computation. As a result, batch GP-based continuous occupancy mapping is restricted to small areas in comparison to the nowadays needs of autonomous robotic systems. We first explain the batch GPOM technique and then develop the algorithm to formulate I-GPOM which relies on kD-tree data structures for map storage and approximate nearest neighbor queries (Arya and Mount 1993; Mount and Arya 2006; Bagon 2009).

4.2.4 Batch Mapping

In the previous part of this chapter, we developed the problem formulation and explained the required steps to solve it. In the case that all robot poses and measurements are available, it is possible to build the map at once. The size of the map and the amount of data to process depend on available computational resources. Building the map using all the data has the advantage of considering the correlation between all points at the time of inference. While this is a preferable inference approach, it is clearly not a tractable method for online or incremental robotic navigation tasks. Figure 4.2 illustrates the process of batch map inference where the regression can be decomposed into two separate GPs as explained in 4.2.2. Following a similar strategy used

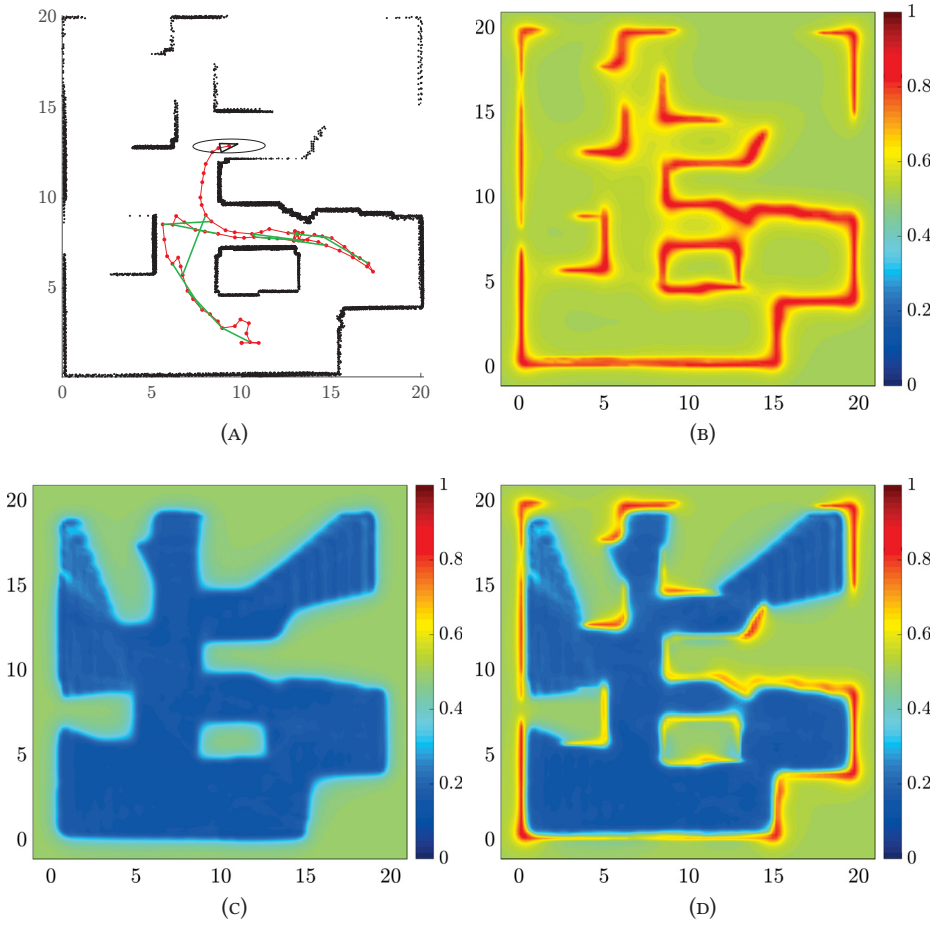


FIGURE 4.4: A regressed continuous occupancy map of the Cave environment, with size of 20 m×20 m. (A) shows the Pose SLAM map; red dots indicate robot poses, green lines depict loop-closures. (B) shows the occupied probability map. (C) shows the unoccupied probability map, and (D) is the overall fused GPOM. The maps in (B) and (C) are built using the Matérn ($\nu = 5/2$) covariance function with isotropic distance measure.

for incremental map building (as discussed in the following subsection), it is plausible to fuse two inferred occupied and unoccupied batch maps into a unique COM, Figure 4.4. We consider batch mapping to be able to compare the performance of the proposed incremental map fusion approach presented in the following subsection.

4.2.5 Map Management and Incremental Mapping

The GP mapper module is shown in Figure 4.5 and takes the processed measurements, i.e. training data, and a test point window centered at the current robot pose as inputs to perform regression and classification steps for local maps generation and fuse them incrementally into the global

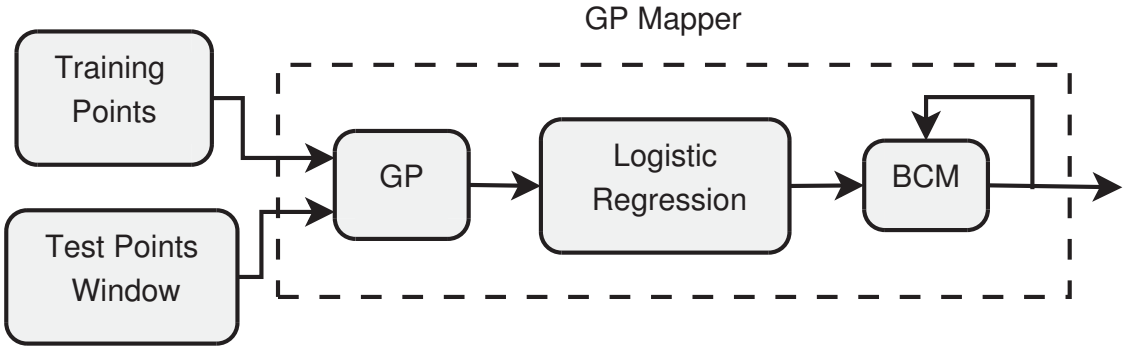


FIGURE 4.5: Schematic illustration of GP Mapper module. GP models the correlation in data and places distributions on test points. The logistic regression classifier squashes the output of GP into probabilities and returns the local map where the BCM module updates the global map incrementally.

frame through the BCM technique (Tresp 2000). An important advantage of a mapping method is its capability to use past information appropriately. The mapping module returns local maps centered at the robot pose. Therefore, to keep track of the global map, a map management step is required where the locally inferred map can be fused with the current global map. This incremental approach allows for handling larger map sizes, and map inference at the local level is independent of the global map. In practice to increase the efficiency of the mapping process, a kD-tree data structure and approximate nearest neighbor search (Arya and Mount 1993; Mount and Arya 2006; Bagon 2009) can be used.

To incorporate new information incrementally, map updates are performed using BCM. The technique combines estimators which were trained on different data sets. Assuming a Gaussian prior with zero mean and covariance Σ and each GP with mean $\mathbb{E}[f_*|\mathcal{D}^{[i]}]$ and covariance $\mathbb{Cov}[f_*|\mathcal{D}^{[i]}]$, it follows that (Tresp 2000)

$$\mathbb{E}[f_*|\mathcal{D}] = \boldsymbol{C}^{-1} \sum_{i=1}^{p_m} \mathbb{Cov}[f_*|\mathcal{D}^{[i]}]^{-1} \mathbb{E}[f_*|\mathcal{D}^{[i]}] \tag{4.9}$$

$$\boldsymbol{C} = \mathbb{Cov}[f_*|\mathcal{D}]^{-1} = -(p_m - 1)(\Sigma)^{-1} + \sum_{i=1}^{p_m} \mathbb{Cov}[f_*|\mathcal{D}^{[i]}]^{-1} \tag{4.10}$$

where p_m is the total number of mapping processes.

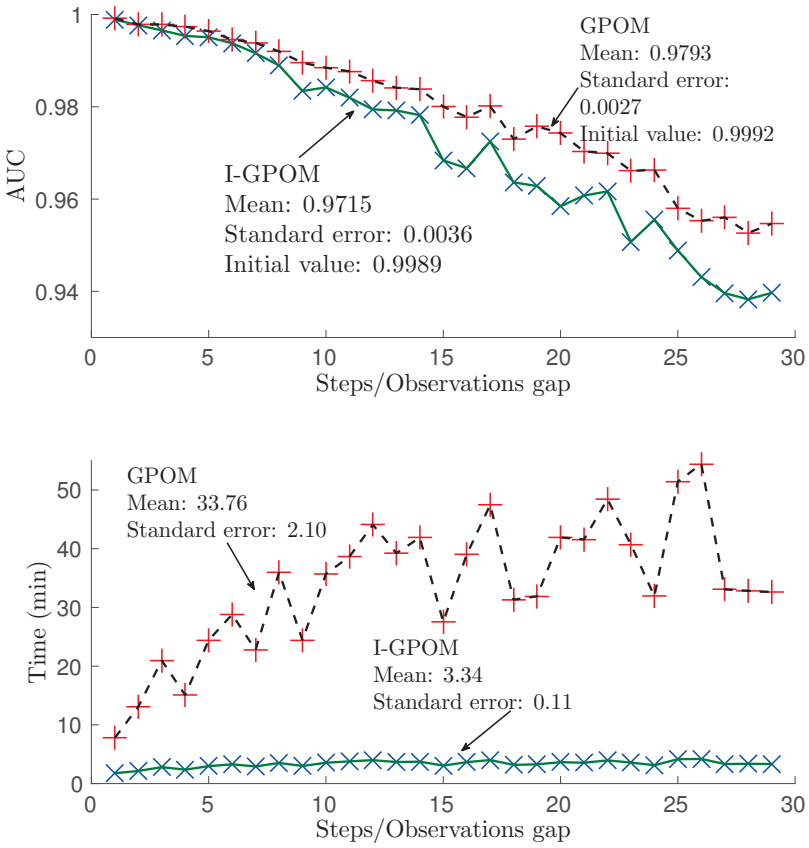


FIGURE 4.6: Comparison of I-GPOM and batch GPOM methods using the Intel dataset with the observations size of 25 laser scans at each step due to the memory limitation for the batch GP computations. The top plot shows the AUC and the bottom plot depicts the runtime for each step. The horizontal axes indicate observations gaps. As the number of gaps grows from 1 to 29, the batch GP outperforms the incremental method as it can learn the correlation between observations at once, however, with higher computational time. On the other hand, the incremental method in nearly constant time produces a similar average map quality with the mean difference of 0.0078.

4.2.6 Mapping Results

A comparison of the incremental (I-GPOM) and batch (GPOM) GP occupancy mapping using the Intel dataset (Howard and Roy 2003) with respect to the Area Under the receiver operating characteristic Curve (AUC) and runtime is presented in Figure 4.6. The probability that the classifier ranks a randomly chosen positive instance higher than a randomly chosen negative instance can be understood using the AUC of the classifier; furthermore, the AUC is useful for domains with skewed class distribution and unequal classification error costs (Fawcett 2006). Without loss of generality, an observation size of 25 laser scans had to be set due to the memory limitation imposed by the batch GP computations with a growing gap between successive laser scans from 1

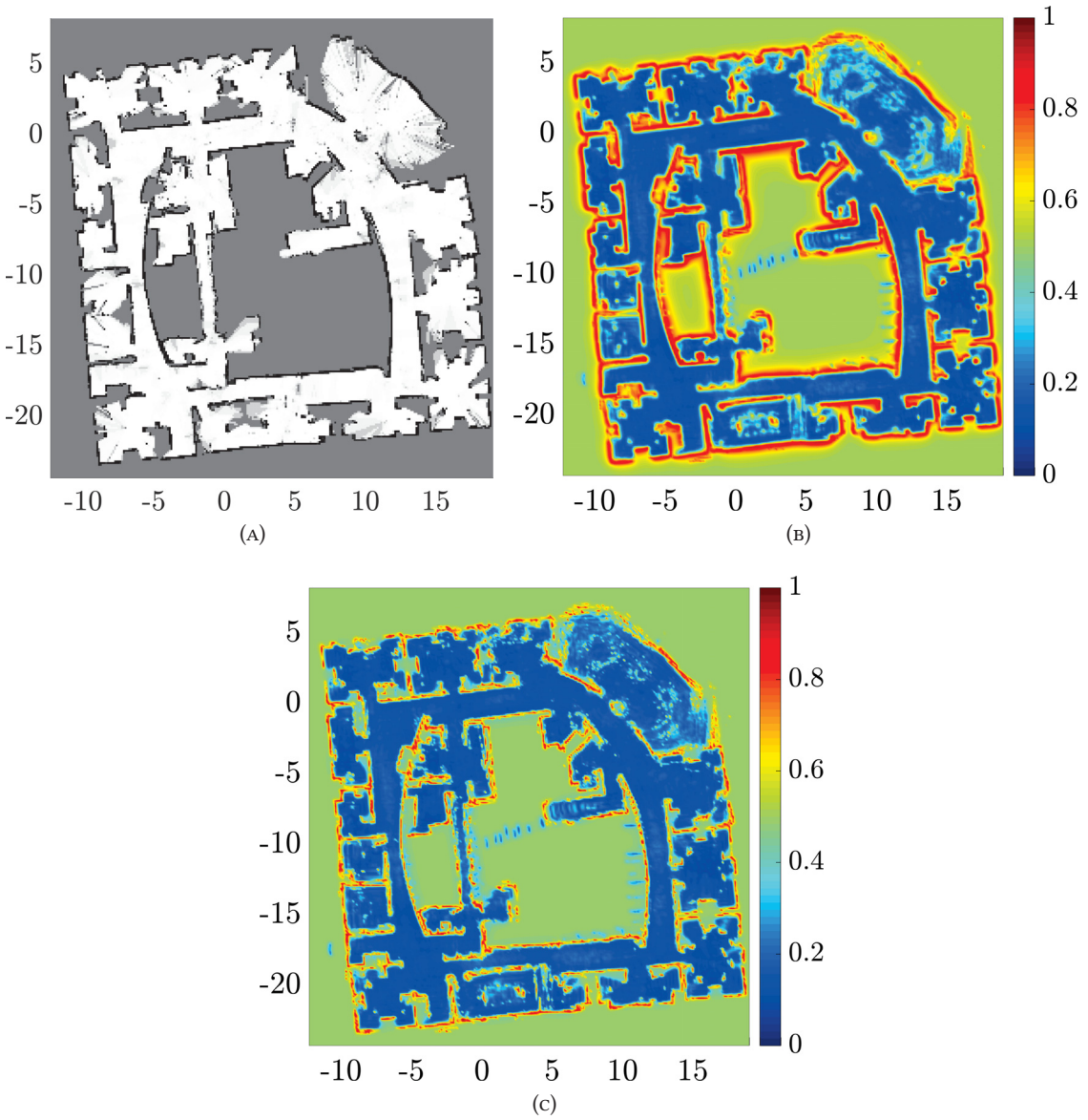


FIGURE 4.7: Occupancy maps visualization; (A) shows the OGM, (B) shows the I-GPOM, and (C) shows the I-GPOM2. The maps are built incrementally using all the available observations in Intel dataset. For the I-GPOM and I-GPOM2 maps the Matérn ($\nu = 3/2$) covariance function is used. I-GPOM and I-GPOM2 can complete partially observable areas, i.e. incomplete areas in the OGM; however, using two GP in I-GPOM2 method produces more accurate maps for navigation purposes. The SLAM problem is solved by using the Pose SLAM algorithm and the map qualities depend on the robot localization accuracy.

to 29. The proposed incremental mapping approach using BCM performs accurately and close to the batch form even with about 8 steps intermission between successive observations and is faster.

The complete results of occupancy mapping with the three different methods in the Intel dataset

Table 4.1: Comparison of the AUC and runtime for OGM, I-GPOM, and I-GPOM2 using the Intel dataset.

Method	AUC	Runtime (min)
OGM	0.9300	7.28
I-GPOM	0.9439	102.44
I-GPOM2	0.9668	114.53

Algorithm 1 `IGPOM()`

Require: Robot pose $\boldsymbol{p}$ and measurements $\boldsymbol{z}$;
1: **if** $firstFrame$ **then**
2: $m = \varnothing$ // Initialize the map
3: optimize GP hyperparameters $\boldsymbol{\theta}$ // Minimize the NLML
4: **end if**
5: $\boldsymbol{X}_* \leftarrow$ `TestDataWindow`$(\boldsymbol{p})$ // Query points grid centered at the robot pose
6: $\boldsymbol{X}_o, \boldsymbol{y}_o \leftarrow$ `Transform2Global`$(\boldsymbol{p}, \boldsymbol{z})$ // Occupied training data, label $+1$
7: $\boldsymbol{X}_f, \boldsymbol{y}_f \leftarrow$ `TrainingData`$(\boldsymbol{p}, \boldsymbol{z})$ // Unoccupied training data, label -1
8: $[\boldsymbol{\mu}_*, \boldsymbol{\sigma}_*] \leftarrow$ `GP`$(\boldsymbol{\theta}, [\boldsymbol{X}_o; \boldsymbol{X}_f], [\boldsymbol{y}_o; \boldsymbol{y}_f], \boldsymbol{X}_*)$ // Compute predictive mean and variance
9: $m \leftarrow$ `UpdateMap`$(\boldsymbol{\mu}_*, \boldsymbol{\sigma}_*, m)$ // Algorithm 4
10: **return** m

are presented in Figure 4.7, while the AUCs are compared in Table 4.1. The I-GPOM2 method demonstrates more flexibility to model cluttered rooms and has higher performance than the other methods. The ground truth map was generated using the registered points map and an image dilation technique to remove outliers. In this way, the ground truth map has the same orientation which makes the comparison convenient. GPOM-based maps infer partially observed regions; however, in the absence of a complete ground truth map, this fact can be only verified using Figure 4.7 and is not reflected in the AUC of I-GPOM and I-GPOM2.

The steps of the incremental GPOM (I-GPOM) are shown in Figure 4.5 and Algorithm 1 where a BCM module updates the global map as new observations are taken. Algorithms 2, 3, 4, and 5 encapsulate the I-GPOM2 methods as implemented in the present work.

4.3 Gaussian Processes Frontier Maps

Constructing a frontier map is the fundamental ingredient of any geometry-based exploration approach. It reveals the boundaries between known-free and unknown areas which are potentially informative regions for map expansion. In contrast to the classical binary representation,

Algorithm 2 `IGPOM2()`

Require: Robot pose $\boldsymbol{p}$ and measurements $\boldsymbol{z}$;
if $firstFrame$ **then**
  $m = m_o = m_f = \varnothing$ // Initialize the map
  optimize GP hyperparameters $\boldsymbol{\theta}_o$, $\boldsymbol{\theta}_f$ // Minimize the NLML
end if
$\boldsymbol{X}_* \leftarrow$ `TestDataWindow`$(\boldsymbol{p})$ // Query points grid centered at the robot pose
$\boldsymbol{X}_o, \boldsymbol{y}_o \leftarrow$ `Transform2Global`$(\boldsymbol{p}, \boldsymbol{z})$ // Occupied training data, label +1
$\boldsymbol{X}_f, \boldsymbol{y}_f \leftarrow$ `TrainingData`$(\boldsymbol{p}, \boldsymbol{z})$ // Unoccupied training data, label −1
$[\boldsymbol{\mu}_{o*}, \boldsymbol{\sigma}_{o*}] \leftarrow$ `GP`$(\boldsymbol{\theta}_o, \boldsymbol{X}_o, \boldsymbol{y}_o, \boldsymbol{X}_*)$ // Compute occupied map predictive mean and variance
$[\boldsymbol{\mu}_{f*}, \boldsymbol{\sigma}_{f*}] \leftarrow$ `GP`$(\boldsymbol{\theta}_f, \boldsymbol{X}_f, \boldsymbol{y}_f, \boldsymbol{X}_*)$ // Compute unoccupied map predictive mean and variance
$m_o \leftarrow$ `UpdateMap`$(\boldsymbol{\mu}_{o*}, \boldsymbol{\sigma}_{o*}, m_o)$ // Algorithm 4
$m_f \leftarrow$ `UpdateMap`$(\boldsymbol{\mu}_{f*}, \boldsymbol{\sigma}_{f*}, m_f)$
$m \leftarrow$ `MergeMap`(m_o, m_f) // Algorithm 5
return m, m_o

Algorithm 3 `FusionBCM`$(\mu_a, \mu_b, \sigma_a, \sigma_b)$

$\sigma_c \leftarrow (\sigma_a^{-1} + \sigma_b^{-1})^{-1}$ // Point-wise calculation of Equation (4.10)
$\mu_c \leftarrow \sigma_c(\sigma_a^{-1}\mu_a + \sigma_b^{-1}\mu_b)$ // Point-wise calculation of Equation (4.9)
return μ_c, σ_c

Algorithm 4 `UpdateMap()`

Require: Global map m, $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and local map m_*, $\boldsymbol{\mu}_*$, $\boldsymbol{\sigma}_*$;
for all $i \in \mathcal{M}_*$ **do**
  $j \leftarrow$ find the corresponding global index of i using a nearest neighbor search
  $\boldsymbol{\mu}^{[j]}, \boldsymbol{\sigma}^{[j]} \leftarrow$ `FusionBCM`$(\boldsymbol{\mu}^{[j]}, \boldsymbol{\mu}_*^{[i]}, \boldsymbol{\sigma}^{[j]}, \boldsymbol{\sigma}_*^{[i]})$ // Algorithm 3
end for
$m \leftarrow$ `LogisticRegression`$(\boldsymbol{\mu}, \boldsymbol{\sigma})$ // Squash data into (0,1), Equation (4.8)
return m

defining frontiers in a probabilistic form using map uncertainty is more suitable for computing expected behaviors. The value of a frontier point can be computed as

$$\bar{f}^{[i]} \triangleq \|\nabla p(m^{[i]})\|_1 - \beta(\|\nabla p(m_o^{[i]})\|_1 + p(m_o^{[i]}) - 0.5) \tag{4.11}$$

where ∇ denotes the gradient operator, and β is a factor that controls the effect of obstacle boundaries. $\|\nabla p(m^{[i]})\|_1$ indicates all boundaries whilst $\|\nabla p(m_o^{[i]})\|_1$ defines obstacle outlines. The subtracted constant is to remove the biased probability for unknown areas in the obstacles probability map.

The frontier surface is converted to a probability frontier map through the incorporation of the

Algorithm 5 `MergeMap()`

Require: Unoccupied map m_f, $\boldsymbol{\mu}_f$, $\boldsymbol{\sigma}_f$ and occupied map m_o, $\boldsymbol{\mu}_o$, $\boldsymbol{\sigma}_o$;
1: **for** all $i \in \mathcal{M}$ **do**
2: $\quad \boldsymbol{\mu}^{[i]}, \boldsymbol{\sigma}^{[i]} \leftarrow$ `FusionBCM`$(\boldsymbol{\mu}_o^{[i]}, \boldsymbol{\mu}_f^{[i]}, \boldsymbol{\sigma}_o^{[i]}, \boldsymbol{\sigma}_f^{[i]})$ // Algorithm 3
3: **end for**
4: $m \leftarrow$ `LogisticRegression`$(\boldsymbol{\mu}, \boldsymbol{\sigma})$ // Squash data into (0,1), Equation (4.8)
5: **return** m

Algorithm 6 `BuildFrontierMap()`

Require: Current map m, $\boldsymbol{\sigma}$ and occupied map m_o, $\boldsymbol{\sigma}_o$;
1: // Compute boundaries
2: $dm \leftarrow \|\nabla p(m)\|_1$, $dm_o \leftarrow \|\nabla p(m_o)\|_1$
3: $\sigma_{min} \leftarrow \min(\boldsymbol{\sigma})$
4: $f \leftarrow \varnothing$
5: // Compute probabilistic frontiers
6: **for** all $i \in \mathcal{M}$ **do**
7: $\quad \bar{f}^{[i]} \leftarrow dm^{[i]} - \beta(dm_o^{[i]} + m_o^{[i]} - 0.5)$
8: $\quad w_f^{[i]} \leftarrow \gamma_f$ sqrt$(\sigma_{min}/\boldsymbol{\sigma}^{[i]})$ // Logistic regression weights
9: $\quad f^{[i]} \leftarrow (1 + \exp(-w_f^{[i]} \bar{f}^{[i]}))^{-1}$ // Squash data into (0,1), Equation (4.12)
10: **end for**
11: **return** f

map uncertainty. To squash the frontier and variance values into the range $[0, 1]$, a logistic regression classifier with inputs from $\bar{f}^{[i]}$ and map uncertainty $\sigma^{[i]}$ is applied to data which yields

$$p(f^{[i]} | m^{[i]}, w_f^{[i]}) = \frac{1}{1 + \exp(-w_f^{[i]} \bar{f}^{[i]})} \tag{4.12}$$

where $w_f^{[i]} = \gamma_f \sqrt{\lambda^{[i]}}$ denotes the required weights and $\gamma_f > 0$ is a constant to control the sigmoid shape. The details of the frontier map computations are presented in Algorithm 6. Figure 4.8 (right) depicts an instance of the frontier map from an exploration experiment in the Cave environment.

In practice, the following steps are required to use the frontier map and check the termination condition:

1. The probabilistic frontier map is converted to a binary map using a pre-defined threshold. Note that any point with a probability higher than 0.5 is potentially a valid frontier.

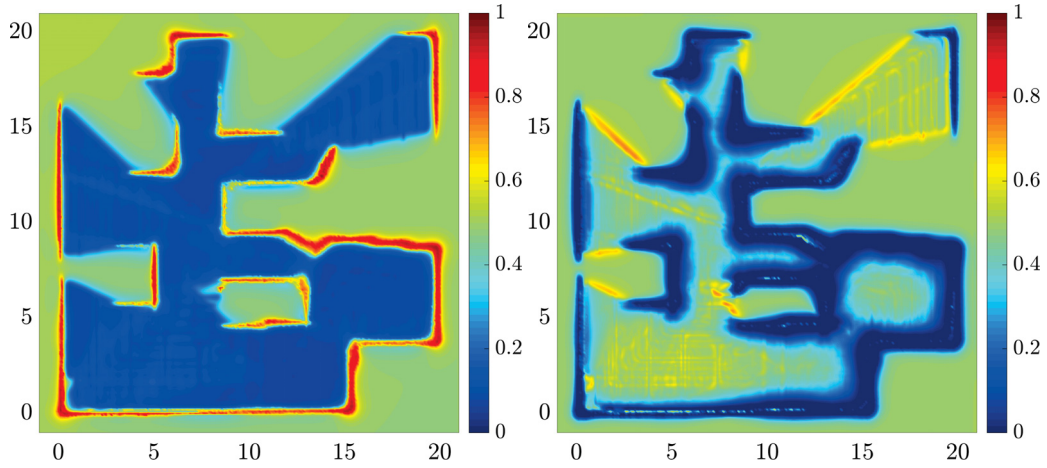


FIGURE 4.8: Inferred continuous occupancy map (left); associated probabilistic frontier map (right). The frontier map highlights the informative regions for further exploration by assigning higher probabilities to frontier points. The lower probabilities show the obstacles and walls whilst the values greater than the "no discrimination" probability, 0.5, can be considered as frontiers.

2. The binary map of frontiers is clustered into subsets of candidate macro-actions (see Chapter 5 for the definition).

3. The centroids of clusters construct a discrete action set at time step t, i.e. $\mathcal{A}_t$, that is used in the utility maximization step.

4. The robot plans a path to each centroid (macro-action) to check its reachability. A centroid that is not reachable is then removed from the action set.

5. The exploration mission continues while the action set $\mathcal{A}_t$ is not empty (repeats from step 1).

4.4 Chapter Summary

In this chapter, we formulated the GP occupancy mapping problem and presented algorithmic implementations of the solution using both batch and incremental techniques. In particular, we showed that the performance of our incremental mapping technique is comparable to that of the batch method and is faster. To make the I-GPOM suitable for geometric-based exploration scenarios, we proposed to use I-GPOM2. We also developed a probabilistic representation of the

geometric frontiers using GPOM. In the next chapter, we employ these maps for autonomous robotic exploration.

Chapter 5

Exploration using Gaussian Processes Maps

In this chapter, we study the problem of autonomous robotic exploration using the developed incremental GPOM techniques in the previous chapter. The classical solution for this task is a geometric-based approach in which the goals represent boundaries of unknown regions of the map. These boundaries are known as frontiers and in the traditional grid-based approach they are represented using a binary map, i.e. there is no notion of uncertainty. To mitigate this deficiency, information gain-based methods that minimize entropy-based cost functions are proposed (Feder et al. 1999; Bourgault et al. 2002; Stachniss et al. 2005; Carlone et al. 2010; Amigoni and Caglioti 2010). Occupancy grid mapping techniques represent the environment by relying on the assumption of independence between cells. This assumption leads to a map posterior inference through marginalization and ignoring structural correlations in the environment. In contrast, GP-based occupancy maps (T O'Callaghan and Ramos 2012; Kim and Kim 2013a) capture structural dependencies of the environment. Therefore, such maps are better suited for accurately computing the statistical properties such as entropy and mutual information.

Decision making is the core part of any robotic exploration algorithm and relies heavily on the chosen utility function. Many cost/utility functions have been examined to improve the overall performance of the exploration algorithms (Juliá et al. 2012). Mutual Information (MI) is a

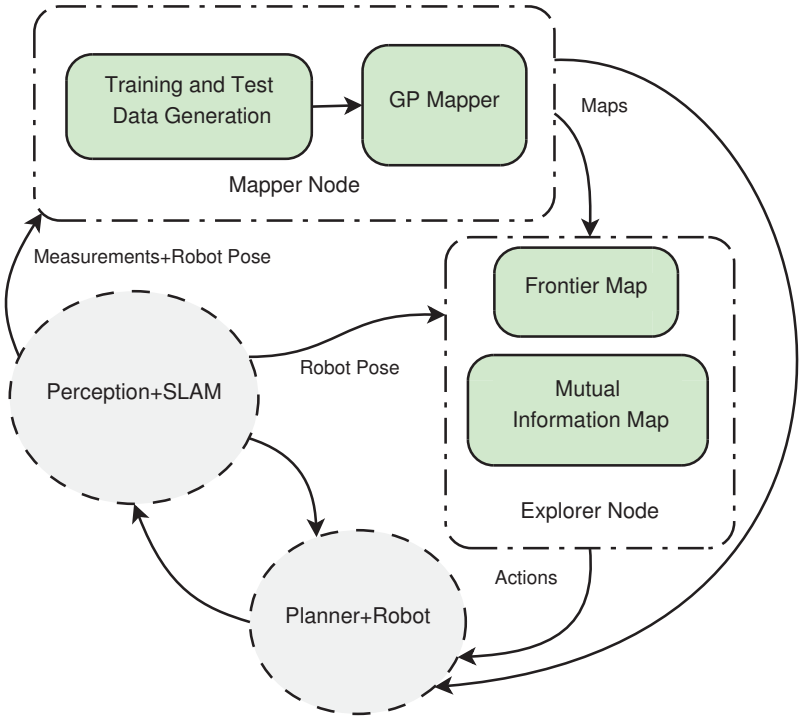


FIGURE 5.1: Schematic illustration of the autonomous mapping and exploration process using GPs maps. The GP mapper module provides the continuous occupancy map which can be exploited to extract geometric frontiers and mutual information maps. The maps also give support to the planner module for basic navigation tasks as well as cost-aware planning. The explorer node returns a macro-action (chosen frontier) that optimizes the expected utility function. The gray nodes are not investigated.

measure of the value of information that quantifies the information gain from sensor measurements (Krause and Guestrin 2005). Therefore, it is a reasonable choice for the utility optimization problem. We develop a greedy mutual information-based exploration strategy that exploits the developed incremental GPOM and probabilistic frontier map techniques for solving the robotic exploration problem. Even though the technique is greedy, it takes into account all possible future measurements. The MI-based utility function is computed at the centroids of geometric frontiers and the frontier with the highest information gain is chosen as the next-best "macro-action". We borrow the notion of macro-action from planning under uncertainty (He et al. 2010) and define it as follows.

Definition 5.1 (Macro-action)**.** *A macro-action is an exploration target (frontier) which is assumed to be reachable through an open-loop control strategy.*

Figure 5.1 depicts the proposed navigation process concept using GPs maps. The proposed technique is in particular interesting due to its simplicity to implement and its strength for dense map

entropy reductions. The technique uses a probabilistic frontier representation and continuous occupancy maps and can handle sparse observations. Furthermore, the MI is computed accurately using a forward sensor model map prediction algorithm. Given the robot pose and corresponding laser scans, GPOM and its associated variance surface are used for generating MI-based macro-actions. The employed measurement model is a standard beam-based mixture model for range-finder sensors (Thrun et al. 2005), however, the proposed algorithm can be adapted to other sensor modalities with reasonable probabilistic observation models.

5.1 Decision Making Problem

The Maximum Expected Utility (MEU) principle states that the robot should choose the action that maximizes its expected utility, in the current state (Russell and Norvig 2009, page 483). The expectation is taken due to the stochastic nature of the state and observations. For a large sequence of actions and continuous state and action space, the computational time of calculating the expected utility function can be exponential. This is often the case in robotics when one tries to calculate the information gain. In the following subsection, we discuss the problem of sequential decision making; then we define the utility functions that are used in the experiments in this chapter.

5.1.1 Sequential Decision Making

In robotics, if the robot poses $\boldsymbol{x}_{1:t}$ are known, i.e. full observability, the sequential decision making problem is an instance of Markov Decision Processes (MDPs). In the MDP framework, the sensor model $p(z|x, m)$ is deterministic and bijective. However, uncertainty in action is allowed (Thrun et al. 2005). In the fully probabilistic case, the state is not fully observable. Therefore, measurements and actions are both stochastic. This is the common case in robotics and the general problem is known as POMDPs (Simmons and Koenig 1995; Kaelbling et al. 1998; Roy et al. 1999; Pineau et al. 2003; Spaan and Vlassis 2005; Roy et al. 2005). In two extreme cases, the *planning horizon* is 1 or ∞. The former case is called *greedy* exploration (or planning), and the latter is often treated as a *discounted* problem in which as the horizon goes to infinity the *payoff* diminishes. Alternatively, a fixed or variable (and bounded) planning horizon can be set according to

the *budget* (Indelman et al. 2015). The budget needs to be defined for a specific problem, e.g. the uncertainty upper bound in the system or the available amount of fuel.

5.1.2 Exploration Policies

The definition of utility/cost function depends on the specific application and aim. In the context of autonomous robotic mapping, typically, the main goal is map completion while maintaining the localization accuracy at an acceptable level [1]. It is also important that the exploration policy not encourage actions that lead to inconsistencies in the SLAM algorithm. Let a_t be an action from the set of all possible actions $\mathcal{A}_t$ [2] at time t. Taking a_t leads to the future measurement Z_{t+1} whose realization is unknown at time t. The exploration (planning) problem is to find the action(s) that results in the measurements which optimize the desired utility/cost function. In the following, we define three most common utility functions for the single robot exploration case.

5.1.2.1 Nearest Frontier

The nearest frontier policy (Yamauchi 1997), as its name implies, it is a cost function that drives the robot towards the closest frontier to its current pose. This technique is greedy and the robot solely considers the cost of navigation. Geometric frontiers can be extracted by a desired method from the occupancy map (Keidar and Kaminka 2014). For the GPOM technique, we use the probabilistic frontier map developed in Section 4.3. Let $\mathcal{F}_t$ be the finite set of all detected frontiers at time t. Let the action a_t be the planned path from the current robot pose to the frontier f_t. The cost function, $f_c : \mathcal{A}_t \to \mathbb{R}_{\geq 0}$, is the length of the path from the current robot pose to the corresponding frontier. Thus the problem can be defined as finding the action that minimizes the cost function. Therefore,

$$a_t^\star = \underset{a_t \in \mathcal{A}_t}{\operatorname{argmin}} \; f_c(a_t) \tag{5.1}$$

In practice, frontier cells/points are clustered, and only those with the size above a threshold are valid. The centroid of each cluster is considered as the target point for path planning.

[1] The required localization accuracy is subject to the specific application.

[2] $\mathcal{A}_t$ can be discrete or continuous.

Remark 5.2. *The cost function in Equation* (5.1) *is modular. It means if* $\mathcal{P}_a$ *and* $\mathcal{P}_b$ *be two partial trajectories and* $\mathcal{P}_{ab}$ *the trajectory constructed by concatenating them, we have* $f_c(\mathcal{P}_{ab}) = f_c(\mathcal{P}_a) + f_c(\mathcal{P}_b)$.

Remark 5.3. *In general, the path length can be seen as the line integral of the curve with the current robot pose and the frontier as its end points. Thus, one can define a scalar field over the map and calculate the cost as the line integral of the scalar field using a Riemann sum. In Equation* (5.1) *the integrand (the scalar field function) is simply* 1.

Remark 5.4. *As it can be seen in Equation* (5.1)*, this method is passive meaning it does not consider the possible return for any future measurements. In an environment with a rich flow of sensory information, this shortcoming is insignificant. However, in general, it can cause severe difficulties for any estimation module relying on sequential informative observations* [3].

5.1.2.2 Information Gain

Intuitively, we are interested to know the amount of information the robot can gain from a measurement before actually taking that measurement. Since future measurements are unknown, we treat them as random. Therefore, by taking the expectation over random future measurements, we can predict the information gain. In the Bayesian framework, this can be interpreted as the distance between the prior and posterior distributions. The KLD presented in Equation (3.17) measures this distance. Taking the expectation over both the state and future measurements results in the information gain (mutual information), Equation (3.19), repeated here for convenience

$$I(X;Y) = D(p(x,y) \parallel p(x)p(y)) = \mathbb{E}_{p(x,y)}[\log \frac{p(x,y)}{p(x)p(y)}]$$

Remark 5.5. *Note that using the definition of the conditional probability, we can write Equation* (3.19) *as*

$$I(X;Y) = \mathbb{E}_{p(x,y)}[\log \frac{p(y|x)}{p(y)}] = \mathbb{E}_{p(x,y)}[\log \frac{p(x|y)}{p(x)}] \tag{5.2}$$

and depending on the specific problem it can simplify the calculation of the MI. However, the expectation is still with respect to the joint distribution of the random variables. Equation (5.2) *also implies that* $I(X;Y) = I(Y;X)$*, a property of MI.*

[3] The problem of selecting an action that results in the best feasible future measurement is known as *active perception.*

Let $p(\boldsymbol{x}_{1:t}, m|\boldsymbol{z}_{1:t})$ be the state estimate of the robot trajectory and the map up to time t. The predictive posterior distribution can then be considered as $p(\boldsymbol{x}_{1:t+1}, m|Z_{t+1}, \boldsymbol{z}_{1:t})$. Note that the subscript t for the map is dropped, as it is assumed the map of the environment is static. To find the action, a_t, that maximizes the information gain-based utility function, $f_I : \mathcal{A}_t \to \mathbb{R}_{\geq 0}$, the problem can be written as

$$a_t^\star = \underset{a_t \in \mathcal{A}_t}{\operatorname{argmax}} \; f_I(a_t) \tag{5.3}$$

In other words, the robot takes the action that leads to the maximum return of information. However, as it is evident from Equation (5.3) the cost of taking that action is not included in the utility function. Therefore, an action with the maximum return of information can be costlier than another action which is less informative.

Remark 5.6. *The mutual information of two random variables X and Y conditioned on a third random variable Z is called conditional mutual information and is denoted by $I(X;Y|Z)$. Although the robot trajectory and the map are conditioned on observations, for the sake of simplicity we refer to the conditional mutual information as mutual information.*

5.1.2.3 Cost-Utility Trade-off

The third approach which is the common case in many practical scenarios is based on the idea of a trade-off between the cost and utility of an action, i.e. the payoff. The total utility function can be constructed by combination of Equations (5.1) and (5.3). The primary problem is that the units of utility/cost functions are different. One solution is the expression of the cost in the form of information loss (uncertainty). Another approach is the combination of them using appropriate coefficients, e.g. a linear combination of the utility and cost functions. Let $g : \mathbb{R}^2_{\geq 0} \to \mathbb{R}_{\geq 0}$ be a function that takes $f_c(a_t)$ and $f_I(a_t)$ as its input arguments. The problem of maximizing the total utility function, $u(a_t) \triangleq g(f_I(a_t), f_c(a_t))$, can then be defined as follows.

$$a_t^\star = \underset{a_t \in \mathcal{A}_t}{\operatorname{argmax}} \; u(a_t) \tag{5.4}$$

5.2 Mutual Information-based Exploration

In information gain-based exploration the utility function is defined to maximize the MI between the current state and future measurements. The expectation over new sets of measurements and actions provides a path and goal which is considered as the "optimal" behavior. The underlying process involves simulating the robot traversing towards a candidate goal and collecting a set of measurements. The widely-employed approach to approximate the expected information gain is using an inverse sensor model through ray casting operation in OGMs (Makarenko et al. 2002; Stachniss et al. 2005; Sim and Little 2006; Valencia et al. 2012; Julian et al. 2014).

Using a continuous representation of the map, we propose to compute the MI of the map and future measurements in the current perception field of the robot at a subset of candidate goals. We take into account all future measurements by taking an expectation over them. We estimate the map posterior with a forward sensor model through the Bayes update formula and compute MI numerically. The GPOM-based maps place distributions over map points, and consequently the computed map entropy using this representation is more descriptive of the real map uncertainty.

5.2.1 Mutual information computation

MI is the reduction in uncertainty of a random variable due to the knowledge of another random variable (Cover and Thomas 1991). In other words, given a measurement $Z = \boldsymbol{z}$ from $\mathcal{Z}$ what will be the reduction in the map $M = m$ uncertainty? The MI between the map and the future measurement $Z_{t+1} = \hat{\boldsymbol{z}}$ is

$$\begin{aligned} I(M; Z_{t+1}|\boldsymbol{z}_{1:t}) &= \int_{\hat{\boldsymbol{z}}\in\mathcal{Z}} \sum_{m\in\mathcal{M}} p(m, \hat{\boldsymbol{z}}|\boldsymbol{z}_{1:t}) \log \frac{p(m, \hat{\boldsymbol{z}}|\boldsymbol{z}_{1:t})}{p(m|\boldsymbol{z}_{1:t})p(\hat{\boldsymbol{z}}|\boldsymbol{z}_{1:t})} d\hat{\boldsymbol{z}} \\ &= H(M|\boldsymbol{z}_{1:t}) - \overline{H}(M|Z_{t+1}, \boldsymbol{z}_{1:t}), \end{aligned} \tag{5.5}$$

where $H(M|\boldsymbol{z}_{1:t})$ and $\overline{H}(M|Z_{t+1}, \boldsymbol{z}_{1:t})$ are map and map conditional entropy respectively, which by definition are

$$H(M|\boldsymbol{z}_{1:t}) = -\sum_{m\in\mathcal{M}} p(m|\boldsymbol{z}_{1:t}) \log p(m|\boldsymbol{z}_{1:t}) \tag{5.6}$$

$$\overline{H}(M|Z_{t+1}, \boldsymbol{z}_{1:t}) = \int_{\hat{\boldsymbol{z}}\in\mathcal{Z}} p(\hat{\boldsymbol{z}}|\boldsymbol{z}_{1:t}) H(M|Z_{t+1} = \hat{\boldsymbol{z}}, \boldsymbol{z}_{1:t}) d\hat{\boldsymbol{z}} \tag{5.7}$$

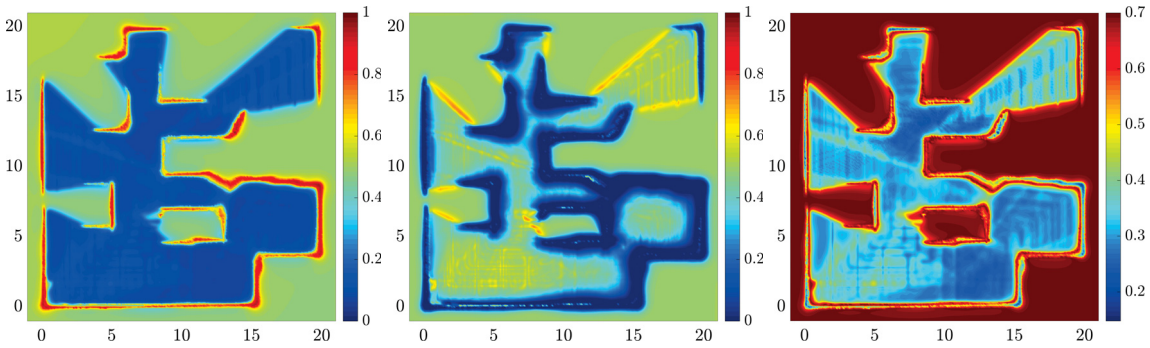


FIGURE 5.2: The inferred continuous occupancy map (left), the associated probabilistic frontier map (middle), and an example of the MI surface. The areas beyond the current perception field of the robot preserve their initial entropy values. The higher values demonstrate regions with greater information gain. The map dimensions are in meters and the MI values in NATS. The robot position is at (10,13), along horizontal and vertical axes, respectively.

To compute the map conditional entropy, the predicted map posterior given the new measurement $Z_{t+1} = \hat{z}_{t+1}$ is required. The Bayesian inference finds the posterior probability for each map point $m^{[i]}$ and k-th beam of the range-finder as

$$p(m^{[i]}|\hat{z}_{t+1}^{[k]}, z_{1:t}) = \frac{p(\hat{z}_{t+1}^{[k]}|m^{[i]})p(m^{[i]}|z_{1:t})}{p(\hat{z}_{t+1}^{[k]}|z_{1:t})} \tag{5.8}$$

$$p(\hat{z}_{t+1}^{[k]}|z_{1:t}) = \sum_{m^{[i]}\in\mathcal{M}} p(\hat{z}_{t+1}^{[k]}|m^{[i]})p(m^{[i]}|z_{1:t}) \tag{5.9}$$

The likelihood function $p(\hat{z}_{t+1}^{[k]}|M = m^{[i]})$ is a beam-based mixture measurement model, where the term $p(\hat{z}_{t+1}^{[k]}|M = 0)$ can be interpreted as the likelihood of not observing the map point at location i, i.e. uniform distribution, and the term $p(\hat{z}_{t+1}^{[k]}|z_{1:t})$ is the marginal distribution over measurements which is calculated in line 14 of Algorithm 7 and denoted by p_z. By numerically integrating over a desired beam range, we can compute the predicted map posterior entropy using Equation (5.7). Note that the conditional entropy does not depend on the realization of future measurements, but it is an average over them.

Let $\mathcal{I}_{t+1}^{[k]}$ be the index set of map points that are in the perception field of the k-th sensor beam at time $t+1$. At any robot location, $\forall i \in \mathcal{I}_{t+1}^{[k]}$, the MI can be written as

$$I^{[i]} = h(m^{[i]}) - \overline{h}(m^{[i]}) \tag{5.10}$$

Algorithm 7 `BuildMIMap()`

Require: Robot pose or desired location, current map estimate m, sensor model;

$\bar{m} \leftarrow m$
// Initialize MI map using current map entropy
$I \leftarrow -(m\log(m) + (1-m)\log(1-m))$
for all k **do**
    Compute $\hat{z}_{t+1}^{[k]}$ and $\mathcal{I}_{t+1}^{[k]}$ using ray casting in m
    for $i \in \mathcal{I}_{t+1}^{[k]}$ **do**
        $\bar{h} \leftarrow 0$ // Map conditional entropy
        for all $z \leq \hat{z}_{t+1}^{[k]}$ **do**
            // Calculate marginal measurement probability p_z
            $p_1 \leftarrow p(\hat{z}_{t+1}^{[k]} | M = 0)$
            $p_2 \leftarrow 0$
            for $j \in \mathcal{I}_{t+1}^{[k]}$ **do**
                $p_1 \leftarrow p_1(1 - m^{[j]})$
                $p_2 \leftarrow p_2 + p(\hat{z}_{t+1}^{[k]} | M = m^{[j]}) m^{[j]} \prod_{l<j} (1 - m^{[l]})$
            end for
            $p_z \leftarrow p_1 + p_2$
            // Map prediction at point i along beam k
            $\bar{m}^{[i]} \leftarrow p_z^{-1} p(\hat{z}_{t+1}^{[k]} | M = m^{[i]}) m^{[i]} \prod_{l<i} (1 - m^{[l]})$
            $\bar{h} \leftarrow \bar{h} + p_z[\bar{m}^{[i]}\log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]})\log(1 - \bar{m}^{[i]})]$
        end for
        $I^{[i]} \leftarrow I^{[i]} + \bar{h} s_z^{-1}$ // Equation (5.10)
    end for
end for
return I

where $h(m^{[i]})$ is the current entropy of the map point $m^{[i]}$ and $\overline{h}(m^{[i]})$ is the estimated map conditional entropy. In practice, at each time step, the map is initialized with the current map entropy, $H(M|\boldsymbol{z}_{1:t})$, and for all map points inside the current perception field the estimated map conditional entropy is subtracted from corresponding initial values. In Algorithm 7, the implementation of the MI map is given where s_z denotes the numerical resolution of integration. In Figure 5.2, an estimated MI map during an exploration experiments in the Cave environment (Howard and Roy 2003) is depicted.

5.2.2 Decision making

The resulting MI map shows the expectation for uncertainty reduction in the map at each place. To define a utility function, the frontier map is initially thresholded and, through k-means, clusters of geometric frontiers are extracted as macro-actions. The decision making process is thus reduced to a standard multi-objective utility maximization problem; see Stachniss et al. (2005) for a similar treatment of the problem.

Let each geometric frontier be regarded as a macro-action from the exploration point of view. The action space can thus be defined as $\mathcal{A}_t = \{a_t^{[j]}\}_{j=1}^{n_a}$. We define the utility function as the difference between the total expected information gain predicted at the macro-action a_t, $f_I(a_t)$, and the corresponding path length from the current robot pose to the same macro-action, $f_c(a_t)$, as follows

$$f_I(a_t) \triangleq \sum_{k=1}^{n_z} \sum_{i \in \mathcal{I}^{[k]}} I^{[i]}(a_t) \tag{5.11}$$

$$u(a_t) \triangleq \alpha f_I(a_t) - f_c(a_t) \tag{5.12}$$

where α is a factor to relate information gain to the cost of motion. The optimal action can be found by maximizing the utility function. Therefore,

$$a_t^\star = \underset{a_t \in \mathcal{A}_t}{\operatorname{argmax}}\, u(a_t) \tag{5.13}$$

Note that the expectation over future measurements and path lengths is already incorporated into the information and cost functions. The optimal action $a_t^\star$ directs the robot towards the frontier with the best balance between information gain and travel cost. This greedy action selection is similar to what is known as next-best view planning in the literature (González-Banos and Latombe 2002; Surmann et al. 2003). It lacks active searching for explicit loop closing actions; however, there are two main motivations to use the proposed utility function:

– The prediction step is often computationally expensive and after taking one action the robot and map states change. Therefore, all previous computations become obsolete and need to be recalculated.

– In office-like indoor environments (such as the Intel office floor space map presented in the results section), a mid-range range-finder sensor covers large areas in the vicinity of the robot. Therefore, even in the absence of explicit loop closing actions, as the robot explores it is highly possible to close informative loops.

5.2.3 Map Regeneration

Loop closure during SLAM can change the map significantly. To account for such changes, we reset and learn the occupancy map with all the available data again. To be able to efficiently detect such a drift in the GPOM we measure the Jensen-Shannon Divergence (JSD) (Lin 1991). The generalized JSD for n probability, $p_1, p_2, ..., p_n$, with weights $\pi_1, \pi_2, ..., \pi_n$ is

$$JS_\pi(p_1, p_2, ..., p_n) = H(\sum_{i=1}^{n} \pi_i p_i) - \sum_{i=1}^{n} \pi_i H(p_i) \tag{5.14}$$

where $H(\cdot)$ is the Shannon entropy function and $p(x_i)$ is the probability associated with variable x_i. All weights are set uniformly as all points are equal.

Alternatively, cumulative relative entropy by summing the computed Jensen-Shannon entropy in each iteration shows map drifts over a period and contains the history of map variations. Consequently, the method is less sensitive to small sudden changes.

Remark 5.7. *The main advantage of JSD over KLD, in this case, is that JSD is bounded. As a result, it is more suitable for decision making (Lin 1991).*

5.3 Exploration Results

In this section, we present results using two publicly available datasets (Howard and Roy 2003). In the first scenario, we use the Intel research lab. map which is a highly structured indoor environment. The second scenario is based on the University of Freiburg campus area. The second map is almost ten times larger than the Intel map and is an example of a large-scale environment with open areas.

Figure 5.3: The constructed environment for exploration experiments using the binary map of obstacles from the Intel dataset.

Table 5.1: The compared exploration methods and their corresponding attributes.

	NF	OGMI	GPNF	GPMI
SLAM	Pose SLAM	Pose SLAM	Pose SLAM	Pose SLAM
Mapping	OGM	OGM	I-GPOM2	I-GPOM2
Frontiers	binary	binary	probabilistic	probabilistic
Utility	path length	MI+path length	path length	MI+path length
Planner	A^*	A^*	A^*	A^*

The experiments include comparison among the original nearest frontier (NF) (Yamauchi 1997), MI-based exploration using OGM (OGMI), the natural extension of NF with a GPOM representation (GPNF) (Ghaffari Jadidi et al. 2014), and the proposed MI-based (GPMI) exploration approaches. NF and OGMI results are computed using OGMs while for the GPOM-based methods the I-GPOM2 representation and the probabilistic frontier map proposed in this work are employed. For all the techniques, we use the A^* algorithm to find the shortest path from the robot position to any frontier. The path cost is calculated using the Euclidean distance between map points. Details about the compared methods are described in Table 5.1.

5.3.1 Experimental setup

The environment is constructed using a binary map of obstacles and, for the Intel map, is shown in Figure 5.3. The simulated robot is equipped with odometric and laser range-finder sensors to provide the required sensory inputs for Pose SLAM. The odometric sensor noise covariance is set

Table 5.2: Parameters for frontier and MI maps computations.

Parameter	Symbol	Value
1) Beam-based mixture measurement model:		
Hit std	σ_{hit}	0.03 m
Short decay	λ_{short}	0.2 m
Max range	r_{max}	
– Intel map		4.0 m
– Freiburg map		60.0 m
Hit weight	z_{hit}	0.7
Short weight	z_{short}	0.1
Max weight	z_{max}	0.1
Random weight	z_{rand}	0.1
2) Frontier map:		
Occupied boundaries factor	β	3.0
Logistic regression weight	γ	10.0
Frontier probability threshold	–	
– Intel map		0.6
– Freiburg map		0.55
Frontier cluster size	–	
– Intel map		14
– Freiburg map		3
Number of clusters	–	
– Intel map		20
– Freiburg map		5
3) MI map and utility function:		
No. of sensor beams over 360 deg	n_z	133
Numerical integration resolution	s_z	
– Intel map		$10/3\ \mathrm{m}^{-1}$
– Freiburg map		$1\ \mathrm{m}^{-1}$
Information gain factor	α	
– Intel map		0.1
– Freiburg map		0.5
Occupied probability threshold	p_o	0.65
Unoccupied probability threshold	p_f	
– Intel map		0.35
– Freiburg map		0.4

to $\Sigma_u = \mathrm{diag}(0.1\,\mathrm{m}, 0.1\,\mathrm{m}, 0.0026\,\mathrm{rad})^2$, and the laser range-finder sensor noise is characterized by the diagonal covariance matrix $\Sigma_y = \mathrm{diag}(0.03\,\mathrm{m}, 0.03\,\mathrm{m}, 0.0013\,\mathrm{rad})^2$. Laser beams are simulated through ray-casting operation over the groundtruth map using the true robot pose. In all the presented results, Pose SLAM (Ila et al. 2010) is included as the backbone to provide localization data together with the number of closed loops. Additionally, for each map, Pose SLAM parameters are set and fixed regardless of the exploration method.

The localization Root Mean-Squared Error (RMSE) is computed at the end of each experiment

by the difference in the robot traveled path (estimated and groundtruth poses) to highlight the effect of each exploration approach on the localization accuracy. The required parameters for the beam-based mixture measurement model (Thrun et al. 2005), frontier maps, and MI maps computations are listed in Table 5.2. The sensitivity of the parameters in Table 5.2 is not high and slight variations of them (about 10%) do not affect the presented results.

The implementation has been developed in MATLAB and GP computations have been implemented by modifying the open source GP library in Rasmussen and Williams (2006). During exploration, map drifts occur due to loop-closure in the SLAM process. As it is computationally expensive to process all measurements from scratch at each iteration, a mechanism has been adopted to address the problem. The cumulative relative entropy by summing the computed JSD can detect such map drifts.

Each technique is evaluated based on six different criteria, namely, travel distance, mapping and planning time, Map Entropy Rate (MER), AUC of the GP occupancy map calculated at the end of each experiment using all available observations, localization RMSE, and the Number of Closed Loops (NCL). Note that none of the compared exploration strategies explicitly plans for loop-closing actions. For each dataset, the results are from 10 independent runs using the same setup and parameters.

5.3.2 Exploration results in the Intel map

An example of the exploration results using GPMI is shown in Figure 5.4. The statistical summary of the results are depicted in Figure 5.5. The most significant part of the results is related to the map entropy rate in which a negative value means the map entropy has been reduced at each step. In the nearest frontier techniques there is no prediction step regarding map entropy reduction; therefore, the results are purely based on chance and structural shape of the environment. OGMI shows marginal improvements over NF with roughly similar computational times for the exploration mission. Thus, it is the preferred technique in comparison with NF.

GPNF and GPMI exploit I-GPOM2 for mapping, exploration, and planning. GP-based methods handle sparse sensor measurements by learning the structural dependencies (spatial correlations) present in the environment. The significant increase in the map entropy rate is due to this fact.

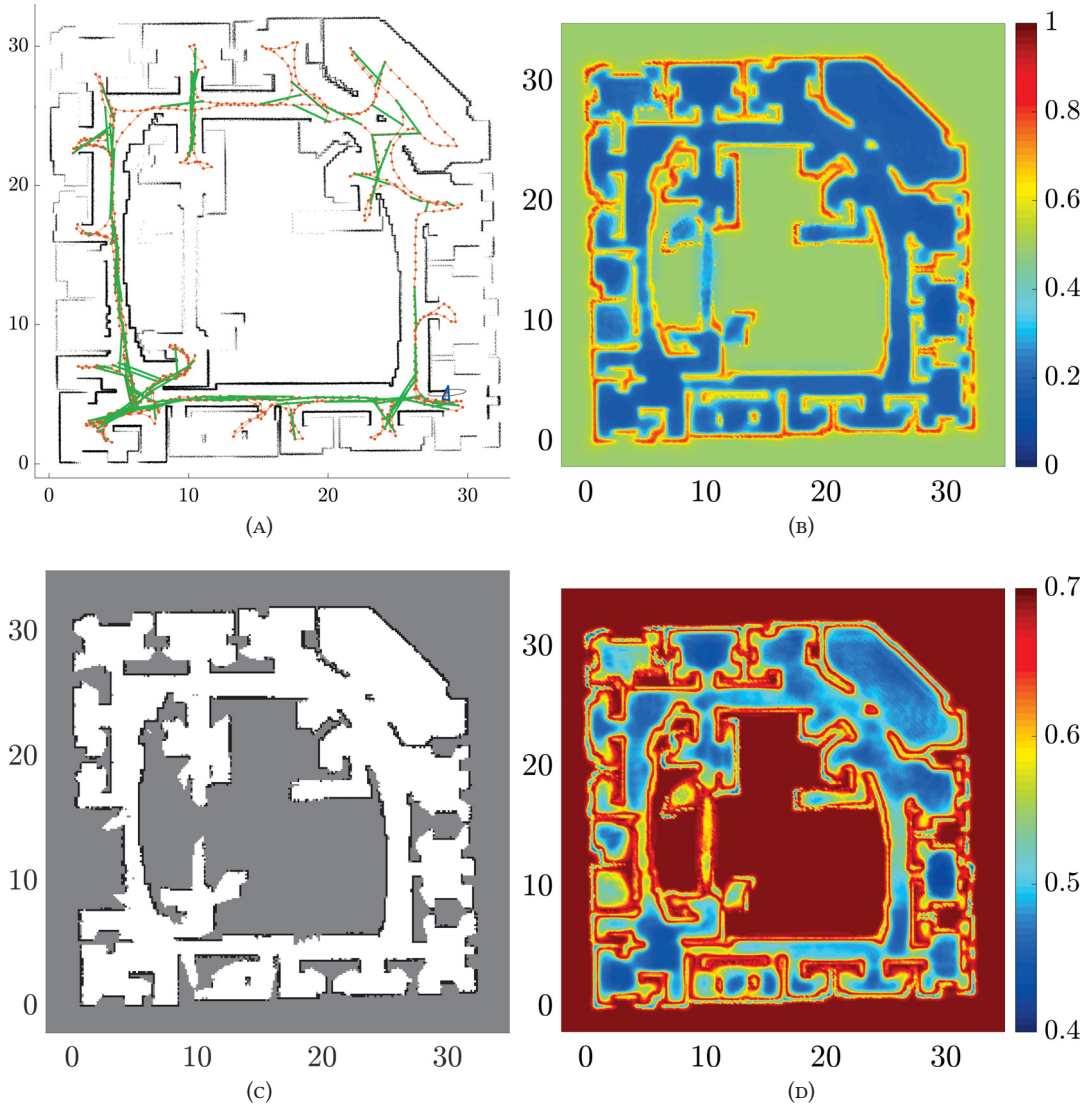


FIGURE 5.4: MI-based exploration in the Intel map. (A) shows Pose SLAM map, (B) shows I-GPOM2, (C) is the equivalent OGM, and (D) is the corresponding entropy map of (B) in NATS. The continuous occupancy map shows the occupancy probability at each location where mid-probability value of 0.5 represents unknown points. The sparse observations due to the occluded perception field in a complex environment such as the Intel map signifies the capabilities of OGM and GPOM methods to cope with such limitations. Map dimensions are in meters. The starting robot position is at (18,26), horizontally and vertically, respectively, and the robot terminates the exploration mission at the most bottom right room.

The results from GPMI show higher travel distance and a higher number of closed loops which can be understood from the fact that information gain in the utility function drives the robot to possibly further but more informative targets. As this behavior does not show any undesirable effect on the localization accuracy, it can be concluded that it performs better than the other techniques; however with a higher computational time. The information gain calculation could

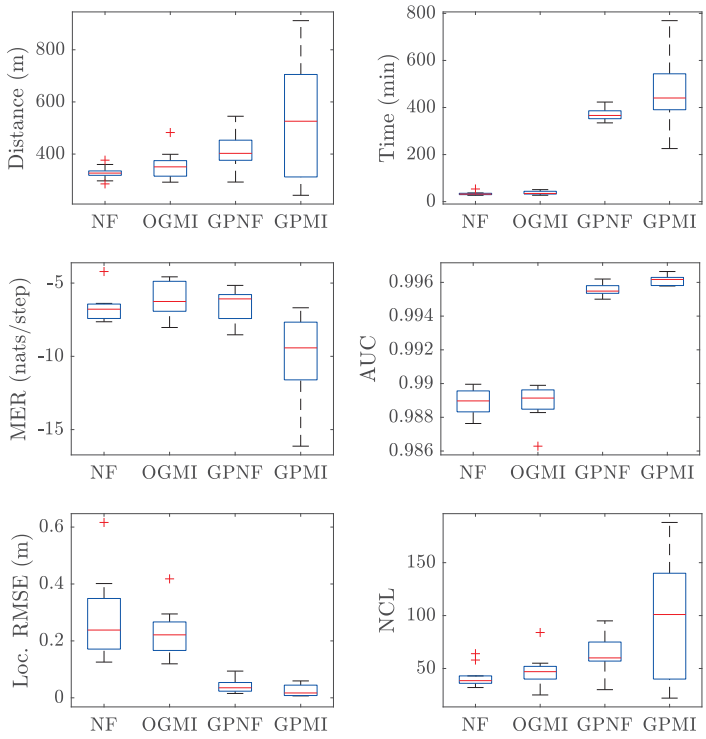


FIGURE 5.5: The box plots show comparison of different exploration strategies in the Intel dataset from 10 independent runs. The compared criteria are travel distance (m), time (min), map entropy rate (nats/step), the mapping performance using the area under the receiving operating characteristic curve, localization root mean-squared error (m), and the number of closed loops by Pose SLAM.

be sped up by using CSQMI due to its similar behavior to MI (Charrow et al. 2015). Under the GPMI scheme, the robot chooses macro-actions that balance the cost of traveling and MI between the map and future measurements. Although the utility function does not include the localization uncertainty explicitly, the correlation between robot poses and the map helps to improve the localization accuracy.

5.3.3 Outdoor scenario: Freiburg Campus

In the second scenario, the map is an outdoor area with a significantly larger size (almost ten times). Figure 5.6 shows the satellite map of the area as well as the trajectory that the robot was driven for data collection. Similar to the first experiment, a binary map of the dataset is constructed and used for exploration experiments. The statistical summary of the experiment results is shown in Figure 5.7. To maintain the computational time manageable, the occupancy maps are built with the coarse resolution of 1 m.

FIGURE 5.6: The left picture shows the satellite map of the Freiburg University Campus where the yellow dashed line indicate the robot trajectory. The middle figure shows the corresponding occupancy map of the dataset (Howard and Roy 2003). The right figure shows the corresponding binary map of obstacles used for exploration experiments. Map dimensions are in meters.

Overall, the trend is similar to the previous test, and specifically, the map entropy rate plot shows a significant difference between GPMI and the other techniques. Again, this significant map entropy rate improvement has been achieved without any undesirable effects on the localization accuracy. Although the GMPI localization error median is slightly higher than NF and GPNF, the overall distribution is sharper. The sharpness of the localization error distribution can be seen as the reliability and repeatability characteristic of GPMI. Since this map has large open areas relative to the robot's sensing range, it is highly unlikely that the robot closes loops by chance. For the GPMI, the number of closed loops has a higher median which supports the idea of implicit loop-closing actions due to the correlations between the map and the robot pose. However, the of NCL distribution has wider tails which does not support its repeatability.

Figure 5.8 shows the results from an exploration run in Freiburg campus map using NF, OGMI, GPNF, and GPMI. The robot behavior is distinguishable in all four maps. In NF case, the robot tends to travel to every corner in the map to complete the partially observable parts of the map. This behavior leads to trajectories along the boundaries of the map. In OGMI, the prediction of the information gain reduces this effect. However, the OGM requires a higher number of measurements to cover an area; therefore, the robot still needs to travel to the corners. In GPNF case, this effect has been alleviated since the the continuous mapping algorithm can deal with sparse measurements. However, in GPMI case, the robot behaves completely different as by taking the expectation over future measurements (calculating MI) the robot does not act based on the current map uncertainty minimization, but improving the future map state in expectation.

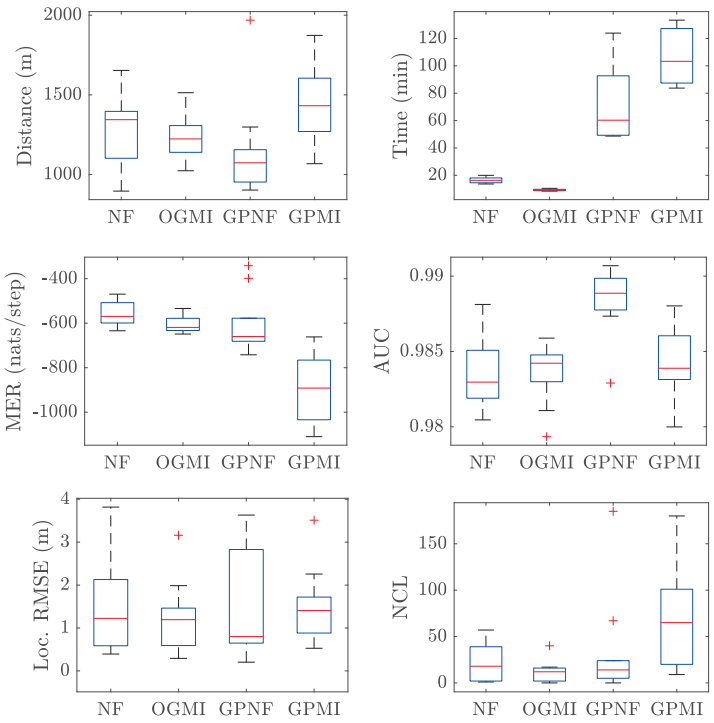


FIGURE 5.7: The box plots show comparison of different exploration strategies in the Freiburg campus dataset from 10 independent runs. The compared criteria are travel distance (m), time (min), map entropy rate (nats/step), the mapping performance using the area under the receiving operating characteristic curve, localization root mean-squared error (m), and the number of closed loops by Pose SLAM.

5.3.4 Computational complexity

For the mapping algorithms, the computational cost of GPs is $\mathcal{O}(n_t^3)$, given the need to invert a matrix of the size of training data, n_t. BCM scales linearly with the number of map points, n_m. The overall map update operation involves a nearest neighbor query for each test point, n_q, and the logistic regression classifier is at worst linear in the number of map points resulting in $\mathcal{O}(n_t^3 + n_q \log n_q + n_m)$. For MI surface, the time complexity is at worst quadratic in the number of map points in the current perception field of the robot, n_p, and linear in the number of sensor beams, n_z, and numerical integration's resolution, s_z, resulting in $\mathcal{O}(n_p^2 n_z s_z)$.

A more sophisticated approximation approach can reduce the computational complexity further. The fully independent training conditional (FITC) (Snelson and Ghahramani 2006) based on inducing conditionals suggests an $\mathcal{O}(n_t n_i^2)$ upper bound where n_i is the number of inducing points. More recently, in Hensman et al. (2013), the GP computation upper bound is reduced to $\mathcal{O}(n_i^3)$ which brings more flexibility in increasing the number of inducing points.

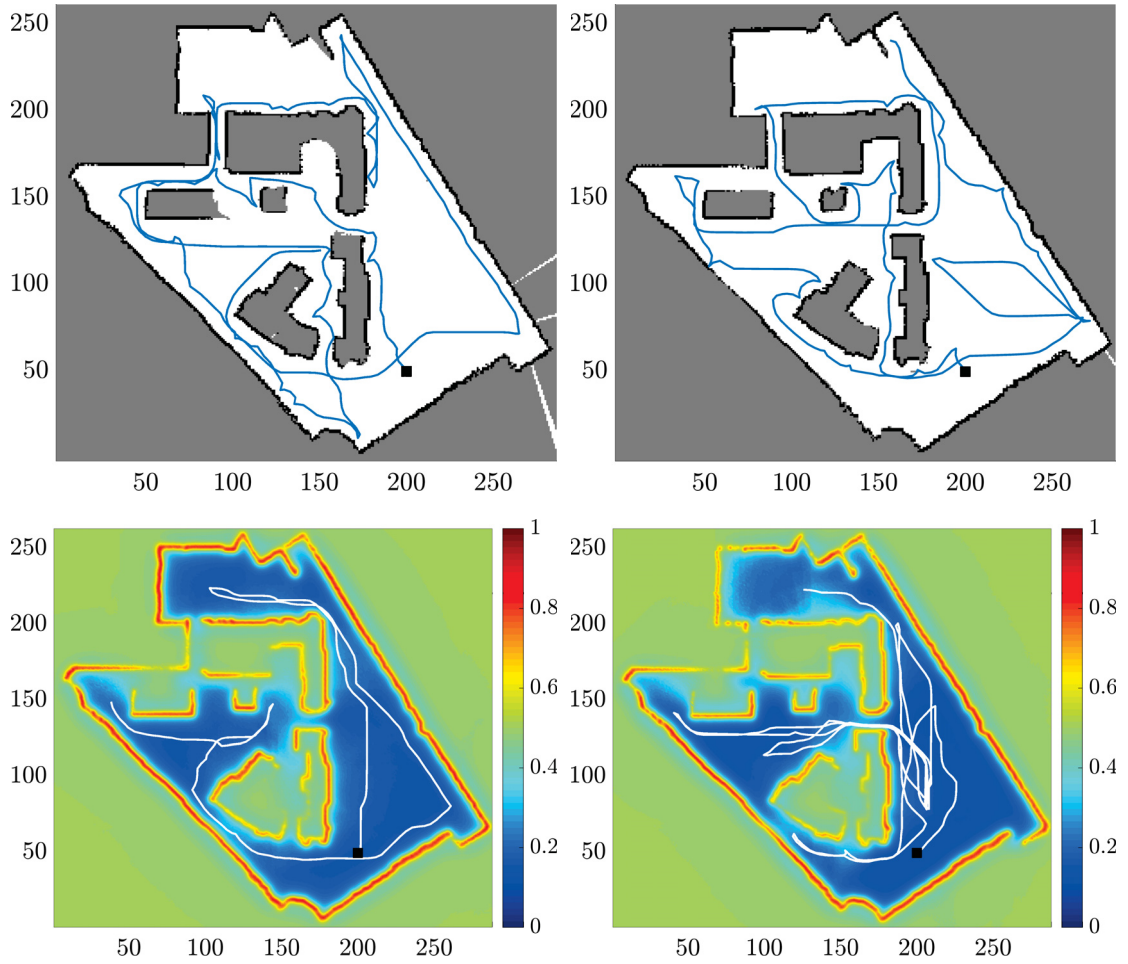


FIGURE 5.8: Illustrative examples of exploration in the Freiburg Campus map. The top left and right, and the bottom left and right figures show the results for NF, OGMI, GPNF, and GPMI, respectively.

5.4 Chapter Summary

In this chapter, we studied the problem of autonomous mapping and exploration for a range-sensing mobile robot using Gaussian processes maps. The continuity of GPOMs is exploited for a novel representation of geometric frontiers, and we showed that the GP-based mapping and exploration techniques are a competitor for traditional occupancy grid-based techniques. The primary motivations stemmed from the fact that high-dimensional map inference requires fewer observations to infer the map, leading to a faster map entropy reduction. The proposed exploration strategy is based on learning spatial correlations of map points using incremental GP-based regression from sparse range measurements and computing mutual information from

the map posterior and conditional entropy. We presented results for two exploration scenarios including a highly structured indoor map as well as a large-scale outdoor area.

When accurate sensors with large coverage relative to the environment are available, existing SLAM techniques can produce reliable localization without the need for an active loop-closure detection. MI-based utility function proposed in this work is suitable for decision making in such scenarios. The more general form of this problem known as active SLAM requires an active search for loop-closures to reduce pose uncertainties. However, the expansion of the state space to both the robot pose and map results in a computationally expensive prediction problem. In the next chapter, we discuss Gaussian processes continuous occupancy mapping problem extensions.

Chapter 6

Gaussian Processes Occupancy Mapping Extensions

In this chapter, we study extensions to the Gaussian processes continuous occupancy mapping problem. There are two classes of occupancy mapping problems that we particularly investigate. The first problem is related to mapping under pose uncertainty and how to propagate pose estimation uncertainty into the map inference. We develop expected kernel and expected sub-map notions to deal with uncertain inputs. In the second problem, we account for the complication of the robot's perception noise using Warped Gaussian Processes (WGPs) (Snelson et al. 2004). This approach allows for non-Gaussian noise in the observation space and captures the possible nonlinearity in that space better than standard GPs. The developed techniques can be applied separately or concurrently to a standard GP occupancy mapping problem. According to our experimental results, although taking into account pose uncertainty leads, as expected, to more uncertain maps, by modeling the nonlinearities present in the observation space WGPs can improve the map quality.

In many scenarios such as robotic navigation, the robot pose is partially observable; and we have access only to an estimate (noise-corrupted version) of the robot pose, as depicted in Figure 6.1. Under these circumstances, the robot requires to navigate in an uncertain environment (Roy et al. 1999; Prentice and Roy 2009; Valencia et al. 2013; Vallvé and Andrade-Cetto 2015), and the

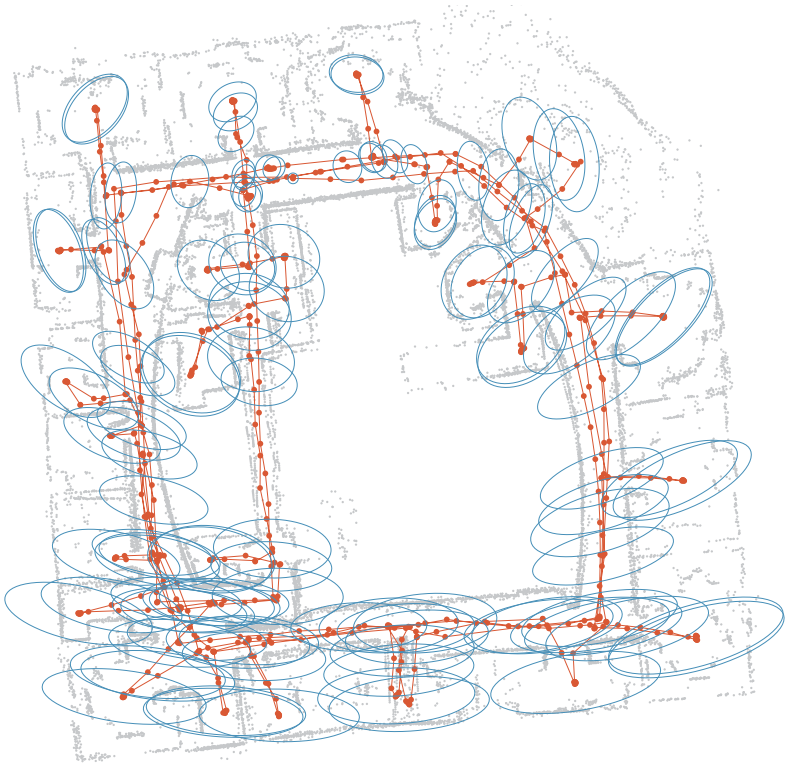


FIGURE 6.1: The pose-graph estimation of the robot trajectory from the Intel research lab. dataset (Howard and Roy 2003), solved using the techniques in Ila et al. (2010); Valencia et al. (2013). The large uncertainties from pose estimation are often ignored in the dense occupancy map representation. For the sake of clarity, loop-closures are omitted, and only one-sixth of the pose covariances are illustrated.

probability distribution of the robot pose will be the input for the mapping problem. In practice, and based on the application, most of the occupancy mapping techniques ignore robot pose uncertainty for map representation; either for efficiency or as the resultant map is not suitable for navigation. Furthermore, dense representation of the state often makes uncertainty propagation intractable. This problem is not unique to *Gaussian Processes Occupancy Maps* (GPOMs), but it is also present in *Occupancy Grid Maps*. With this motivation, we study the problem of GP occupancy mapping under pose uncertainty. The first solution is uncertainty propagation through *kernel* functions. The second solution we propose uses the *expected sub-map* notion to incorporate pose uncertainties into the map building process.

The second problem studied, as discussed in Subsection 4.2.2, is motivated by the fact that due to the smoothness common in the resultant regressed maps, inferring a high-quality map compatible with the actual shape of the environment is non-trivial. Furthermore, for a complicated

task such as robotic mapping (Thrun 2003a), the additive white Gaussian noise assumption in standard GPs can be simplistic. To account for these problems, we improve the incremental GPOM technique using Warped Gaussian Processes Occupancy Maps (WGPOMs). The core idea is to map the target values through a warping (transforming) function to capture the nonlinear behavior of the observations.

We tackle the mentioned problems to improve map quality and provide results using incremental WGPOMs under pose uncertainty.

Remark 6.1. *In this chapter, we develop GPOMs extensions using the I-GPOM. The computational overhead to account for uncertain robot pose can be prohibitive for the case of I-GPOM2, especially in large environments. Further, WGPOM is also an incremental mapping technique.*

6.1 Mapping Under Pose Uncertainty

The main challenge in building occupancy maps under robot pose uncertainty is the dense representation of map belief which makes uncertainty propagation computationally expensive. Maximum likelihood dense map representations are currently the common practice which does not necessarily produce correct maps, especially if pose estimation uncertainties are significant. This popularity can be understood from the fact that employing an environment representation constructed with significant uncertainties results in vague obstacles and free space and is not suitable for robotic motion planning and navigation. However, accounting for pose uncertainties in mapping is not only important for correct map representations, but also for motion planning (prediction) tasks.

6.1.1 Problem Statement and Formulation

Let $\mathcal{M}$ be the set of possible static occupancy maps. We consider the map of the environment as an n_m-tuple random variable $(M^{[1]}, \ldots, M^{[n_m]})$ whose elements are described by a normal distribution $M^{[i]} \sim \mathcal{N}(\mu^{[i]}, \sigma^{[i]})$, $\forall\ i \in \{1 : n_m\}$. Let $\mathcal{X} \subset \mathbb{R}^2$ be the set of spatial coordinates to build a map on. Let $y = \tilde{y} + \epsilon_y$ be a noisy measurement (class label; -1 and 1 for unoccupied and occupied, respectively) at a noisy sample $\boldsymbol{x} = \tilde{\boldsymbol{x}} + \boldsymbol{\epsilon}_x$, where $\epsilon_y \sim \mathcal{N}(0, \sigma_n^2)$ and $\boldsymbol{\epsilon}_x \sim \mathcal{N}(\mathbf{0}, \Sigma_x)$.

Define a training set $\mathcal{D} = \{(\boldsymbol{x}^{[i]}, y^{[i]}) \mid i = 1 : n_t\}$ which consists of noisy measurements at noisy locations. Let function $f : \mathcal{X} \to \mathcal{M}$, i.e. $y = f(\tilde{\boldsymbol{x}} + \boldsymbol{\epsilon}_x) + \epsilon_y$, be the real underlying process that we model as a Gaussian process

$$f(\boldsymbol{x}) \sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}')) \tag{6.1}$$

where $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is the *covariance function* or *kernel*; and $\boldsymbol{x}$ and $\boldsymbol{x}'$ are either in the training or the test (query) sets. Estimate $p(M = m \mid \mathcal{D})$, i.e. the map posterior probability given a noisy training set. For a given query point in the map, $\boldsymbol{x}_*$, GP predicts a mean, μ, and an associated variance, σ. We can write

$$m^{[i]} = y(\boldsymbol{x}_*^{[i]}) \sim \mathcal{N}(\mu^{[i]}, \sigma^{[i]}) \tag{6.2}$$

To show a valid probabilistic representation of the map $p(m^{[i]} \mid \mathcal{D})$, the classification step squashes data into the range $[0, 1]$.

Approximate methods for uncertainty propagation into GPs models through kernel functions are proposed in Girard (2004), and developed for GPOM in O'Callaghan et al. (2010). Generally speaking, training and query points can both be noisy. In Girard (2004), the problem of prediction at an uncertain input is discussed while it is assumed the input in training data is noise free. In O'Callaghan et al. (2010), using a similar approach, the idea is extended to account for noisy training input. Similarly, we assume the input in training data is uncertain and query points are deterministic. In this chapter, we employ this technique and develop the expected sub-map technique for approximate uncertainty propagation, then incorporate both techniques into the WGPOM framework.

We propose two methods to solve the defined problem. The first approach is based on the *expected kernel*. The alternative approach, *expected sub-map*, treats all inputs deterministically and propagates pose uncertainties through uncertain map fusion. The following assumptions are made in the present work:

Assumption 6.2 (Deterministic query points)**.** *In the problem of Gaussian processes occupancy mapping under robot pose uncertainty, query points are deterministic.*

Assumption 6.3. *The covariance function in the expected sub-map method is stationary,* $k(\boldsymbol{x}, \boldsymbol{x}') = k(\|\boldsymbol{x} - \boldsymbol{x}'\|)$.

Remark 6.4. *Using Assumption 6.3, map inference in the local coordinates of the robot (local map) can be done using deterministic inputs.*

6.1.2 Expected Kernel

The core idea in the expected kernel approach is taking an expectation of the covariance function over uncertain inputs. Let $\boldsymbol{x}$ be distributed according to a probability distribution $p(\boldsymbol{x})$. The expected covariance function can be computed as

$$\tilde{k} = \mathbb{E}[k] = \int_{\Omega} k dp \tag{6.3}$$

In general, this integral is analytically intractable; therefore we employ two numerical approximations to solve (6.3). However, for the case of the SE kernel, a closed-form solution exists (Girard et al. 2003). Hence, once the expected covariance matrix is calculated, we can compute the predictive conditional distribution for a single query point similar to standard GPs.

Monte Carlo Integration

Since we assume the distribution of the uncertain input is known, by drawing independent samples, $\boldsymbol{x}^{[i]}$, from $p(\boldsymbol{x})$ and using a Monte-Carlo technique, we can approximate Equation (6.3) by

$$\tilde{k} = \frac{1}{n} \sum_{i=1}^{n} k_i \tag{6.4}$$

where k_i is the covariance function computed at $\boldsymbol{x}^{[i]}$.

Remark 6.5. *In Equation* (6.4)*, the covariance* $k(\boldsymbol{x}, \boldsymbol{x})$ *and cross-covariance* $k(\boldsymbol{x}, \boldsymbol{x}_*)$ *are both denoted as* k_i*. Depending on the input, the integration is only performed on training points as it is assumed query points are deterministic.*

Gauss-Hermite Quadrature

Gauss-Hermite quadrature (Davis and Rabinowitz 1984) of integrals of the kind $\int_{-\infty}^{\infty} \exp(-x^2) f(x) dx$ are given by Equation (3.27), repeated here for convenience

$$\int_{-\infty}^{\infty} \exp(-x^2) f(x) dx \approx \sum_{j=1}^{n} w^{[j]} f(x^{[j]})$$

The multi-variate normal distribution of noisy input is given by $\mathcal{N}(\tilde{\boldsymbol{x}}, \Sigma_x)$. Through a change of variable such that $\boldsymbol{L}\boldsymbol{L}^T = 2\Sigma_x$ and $\boldsymbol{u} = \boldsymbol{L}^{-1}(\boldsymbol{x} - \tilde{\boldsymbol{x}})$, where $\boldsymbol{L}$ is a lower triangular matrix that can be calculated using a Cholesky factorization, the Equation (6.3) can be approximated as

$$\tilde{k} = (2\pi)^{\frac{-d}{2}} \sum_{i_1=1}^{n} \dots \sum_{i_d=1}^{n} \bar{w} k_{i_{1:d}} \tag{6.5}$$

where $\bar{w} \triangleq \prod_{j=1}^{d} w^{[i_j]}$, $u^{[i_j]}$ are the roots of the Hermite polynomial H_n, $\boldsymbol{u}_{i_{1:d}} \triangleq [u^{[i_1]}, \dots, u^{[i_d]}]^T$, and $k_{i_{1:d}}$ is the covariance function computed at $\boldsymbol{x}_{i_{1:d}} = \boldsymbol{L}\boldsymbol{u}_{i_{1:d}} + \tilde{\boldsymbol{x}}$. When $d = 2$, we can simplify Equation (6.5) and write

$$\tilde{k} = \frac{1}{2\pi} \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} \bar{w} k_{i_{1:2}} \tag{6.6}$$

Remark 6.6. *We assumed points from the map spatial support, $\boldsymbol{x} \in \mathcal{X}$, are global coordinates. In practice, to transform local points, an unscented transform (Julier and Uhlmann 1997) is used to reduce linearization errors (O'Callaghan et al. 2010).*

An illustrative example of GP regression where inputs are uncertain is shown in Figure 6.2. By propagating the input uncertainty using the expected kernel, the output does not follow the observations exactly yet remains consistent as the underlying function is within the estimated uncertainty bounds.

6.1.3 Expected Sub-map

As an alternative strategy, We exploit the fact that a stationary covariance function does not depend on the selected coordinates, i.e. the global or a local sub-map frame. Therefore, we treat all training inputs as noise free and conduct map inference using deterministic inputs in the local

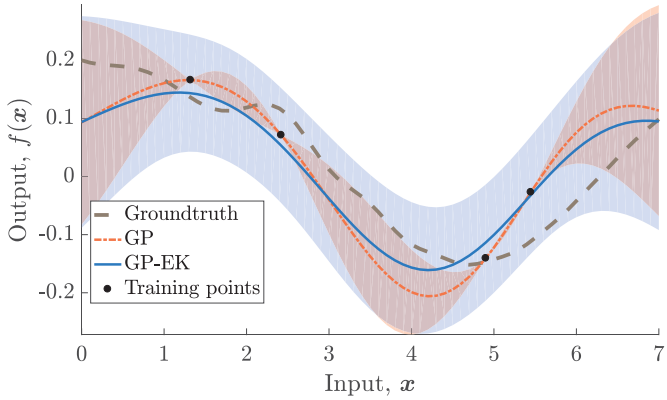


FIGURE 6.2: The plot shows an example of GP regression with uncertain inputs. The GP-EK shows GP regression by incorporating the input uncertainty using the expected kernel. The standard GP results are generated by ignoring the input uncertainty which cannot provide a consistent solution. The ground truth function is $f(x) = \frac{1}{5}\cos(x^2)e^{-x} + \frac{3}{20}\sin(x)$ and training points are corrupted by $\epsilon_x \sim \mathcal{N}(0, 0.6^2)$.

(sensor/robot) frame. To fuse the inferred sub-map into the global map, we draw independent samples from $p(\boldsymbol{x}_t)$. In other words, by taking the expectation over the location of the sub-map, we propagate uncertainty of each map point to its neighborhood. Thus, we have

$$p(m|\tilde{\boldsymbol{x}}, y) = \int p(m|\boldsymbol{x}, y)p(\boldsymbol{x}_t)d\boldsymbol{x}_t \tag{6.7}$$

and by drawing independent samples from $p(\boldsymbol{x})$ and using a Monte-Carlo approximation it follows that

$$p(m|\tilde{\boldsymbol{x}}, y) \approx \frac{1}{n}\sum_{j=1}^{n} p(m_j|\boldsymbol{x}_j, y) \tag{6.8}$$

Note that any sub-map $p(m_j|\boldsymbol{x}_j, y)$ can be fused into the global map using Algorithm 4. However, as a result of sampling, the expected map, $p(m|\tilde{\boldsymbol{x}}, y)$, is similar to a mixture distribution; therefore, the mean and variance calculations need to be addressed accordingly. We present the following proposition to calculate the first two moments of $p(m|\tilde{\boldsymbol{x}}, y)$.

Lemma 6.7 (Sample mean and variance of a mixture distribution). *Let $X_1, X_2, ..., X_n$ be random variables that are distributed according to probability densities $p(x_1), p(x_2), ..., p(x_n)$, with constant weights $w_1, w_2, ..., w_n$, where $\sum_{i=1}^{n} w_i = 1$. The probability density function of the mixture is $p(x) = \sum_{i=1}^{n} w_i p(x_i)$. Given $\mu_i = \mathbb{E}[X_i]$ and $\sigma_i^2 = \mathbb{V}[X_i]$, the mean and variance of the mixture density is*

given by

$$\mu = \mathbb{E}[X] = \sum_{i=1}^{n} w_i \mathbb{E}[X_i] \tag{6.9}$$

$$\sigma^2 = \mathbb{V}[X] = \sum_{i=1}^{n} w_i (\sigma_i^2 + \mu_i^2) - (\sum_{i=1}^{n} w_i \mu_i)^2 \tag{6.10}$$

Proof. The proof follows from the fact that for the k-th moment of the mixture, we can write

$$\mathbb{E}[X^{(k)}] = \sum_{i=1}^{n} w_i \mathbb{E}[X_i^{(k)}] \tag{6.11}$$

and define the variance accordingly. □

Proposition 6.8 (Expected sub-map fusion)**.** *In incremental map building, to compute $p(m|\tilde{\boldsymbol{x}}, y)$, sampled sub-maps can be fused into the global map using the following equations*

$$\mathbb{E}[M] = \frac{1}{n} \sum_{j=1}^{n} \mathbb{E}[M_j] \tag{6.12}$$

$$\mathbb{V}[M] = \frac{1}{n} (\sum_{j=1}^{n} (\mathbb{V}[M_j] + \mathbb{E}[M_j]^2) - \frac{1}{n} (\sum_{j=1}^{n} \mathbb{E}[M_j])^2) \tag{6.13}$$

where $\mathbb{E}[M_j]$ is the updated global map built using the j-th independently drawn robot pose sample, and (6.12) *and* (6.13) *can be computed point-wise for every map point $m^{[i]}$.*

Proof. The proof directly follows from Lemma 6.7. □

Therefore, we can perform incremental map fusion by considering the robot pose uncertainty without modifying the GP framework.

6.2 Warped GP Occupancy Mapping

The primary challenge in modeling the environment "accurately" is the different nature of free and occupied classes. Free space tends to span vast areas while occupied space often represents the structural shape of the environment. In addition, the assumption of additive Gaussian noise

in the observations in standard GPs is unable to capture complexity in observations appropriately. We propose to employ Warped Gaussian Processes to account for the nonlinear behavior of observations. This method is appealing as it allows for non-Gaussian noise in the observation space. However, exact inference is not possible anymore, and approximate inference algorithms such as *expectation propagation* (Minka 2001) or *variational Bayes* (Jordan et al. 1999) are required.

The idea to accommodate non-Gaussian distributions and noise is to use a nonlinear monotonic function for warping (transforming) the observation space (Snelson et al. 2004). Let $g_w(\cdot)$ be a transformation from the observation space to a latent space as

$$t^{[i]} = g_w(y^{[i]}; \boldsymbol{\psi}) \qquad i = 1:n \tag{6.14}$$

where $\boldsymbol{\psi}$ denotes the vector of warping function hyperparameters and $\boldsymbol{t} = [t^{[1]}, \ldots, t^{[n]}]^T$ is the vector of latent targets. Now we can re-write the GP formulation for the latent target and by accounting for the transformation between a true observation and the latent target, the NLML can be written as

$$\begin{aligned} \log\ p(\boldsymbol{y}|X, \boldsymbol{\theta}, \boldsymbol{\psi}) = &-\frac{1}{2} g_w(\boldsymbol{y})^T [\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n]^{-1} g_w(\boldsymbol{y}) \\ &-\frac{1}{2}\log\ |\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n| - \frac{n}{2}\log\ 2\pi + \sum_{i=1}^{n} \log \left.\frac{\partial g_w(y)}{\partial y}\right|_{y^{[i]}} \end{aligned} \tag{6.15}$$

in which the last term is the Jacobian of the defined transformation. To compute the mean at a new test point, it is possible to calculate the expectation of the inverse warping function over the latent target predictive density, therefore

$$\mathbb{E}[y^{[n+1]}] = \int g_w^{-1}(t) \mathcal{N}(\hat{t}^{[n+1]}, \sigma^{[n+1]}) dt = \mathbb{E}[g_w^{-1}] \tag{6.16}$$

This integral can be computed numerically using Gauss-Hermite quadrature with a weighted sum of the inverse warping function g_w^{-1}.

Inspired by the neural network transfer functions, a sum of hyperbolic tangent functions satisfies the requirements for the transformation to be monotonic and at the same time allowing for complicated mappings. With hyperparameters vector $\boldsymbol{\psi} = [\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}]^T$, the function can be defined

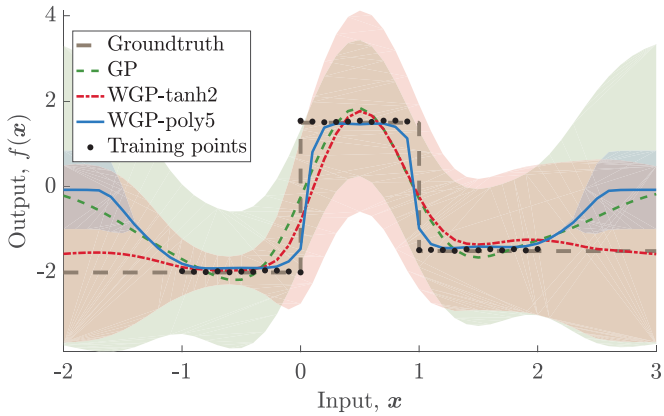


FIGURE 6.3: A challenging example of regression using standard and Warped GPs. The measurements are sampled from the true output values by adding noise, i.e. $\epsilon_y \sim \mathcal{N}(0, 0.05^2)$. The warping functions are $\tanh(\ell = 2)$ and polynomial of degree five. The $\tanh$ demonstrate a better extrapolating behavior, while the polynomial function can follow the underlying function more closely. However, polynomials are prone to over-fitting as it can be seen that the estimated uncertainties (blue shaded region) are significantly narrower.

as

$$g_w(y;\boldsymbol{\psi}) = y + \sum_{i=1}^{\ell} a^{[i]} \tanh(b^{[i]}(y + c^{[i]})) \quad a^{[i]}, b^{[i]} \geq 0 \; \forall i \in \{1:\ell\} \tag{6.17}$$

where the parameter ℓ is the number of steps and has to be set depending on the complexity of observations, $\boldsymbol{a} = [a^{[1]}, \ldots, a^{[\ell]}]$, $\boldsymbol{b} = [b^{[1]}, \ldots, b^{[\ell]}]$, and $\boldsymbol{c} = [c^{[1]}, \ldots, c^{[\ell]}]$. Alternative warping functions can be polynomials ($\boldsymbol{\psi} = \boldsymbol{c}$):

$$g_w(y;\boldsymbol{\psi}) = y + \sum_{i=2}^{\ell} c^{[i-1]} \text{sgn}(y)|y|^i \quad c^{[i]} \geq 0 \; \forall i \in \{2:\ell\} \tag{6.18}$$

Figure 6.3 shows a simple yet challenging example for regression using standard and Warped GPs. The measurements are corrupted by an additive Gaussian noise. Even though the noise is still Gaussian, the complicated structure of the underlying function makes modeling it non-trivial. Note that the inputs are deterministic.

Remark 6.9. *By increasing the number of training points, it is possible to generate more accurate results using standard GPs. However, given the cubic time complexity of GPs, dense training datasets reduce the scalability of the algorithms significantly.*

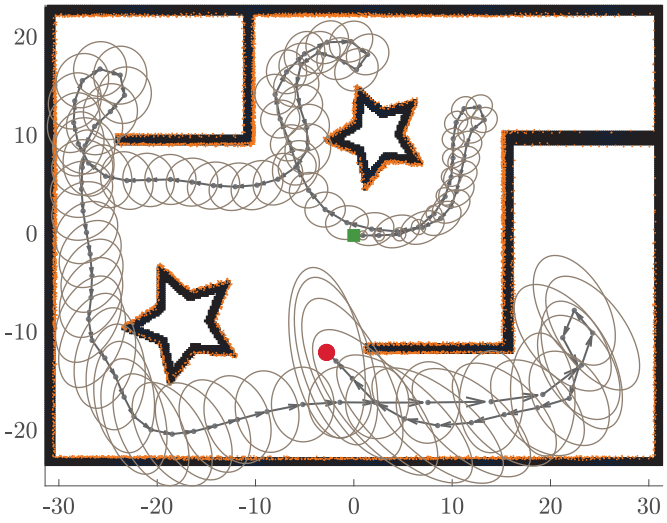


FIGURE 6.4: The synthetic dataset used for comparison of GPOM and WGPOM under various uncertainty propagation conditions. The figure shows collected observations along the robot trajectory (orange dots) and the robot position uncertainty ellipse at each corresponding pose ($\boldsymbol{Q}_3$ scenario). The robot starting position is shown in green. Map dimensions are in meters.

6.3 Results and Discussion

We now present results from experiments using a synthetic dataset and a real publicly available pose-graph dataset. The synthetic dataset, Figure 6.4, is built in such a way as to highlight the strength of WGPOM to model complicated structural shapes and to better appreciate the mapping performance of the incremental GPOM and WGPOM under the Expected Kernel (EK) and Expected Sub-Map (ESM) uncertainty propagation techniques. On the other hand, the Intel dataset (Howard and Roy 2003), as shown in Figure 6.1, exposes an extreme real-world example of the problem at hand where highly uncertain robot poses along the estimated trajectory are present.

We compare the overall taken time to build the entire map (using all the available data) as well as the map accuracy using the AUC (Fawcett 2006). For each model, we learn the hyperparameters at the first increment of map building by minimization of the NLML using the first set of training data with manual supervision to ensure the best possible outcome for all models. The data processing and computations for the incremental map building are implemented using MATLAB.

TABLE 6.1: Details of the experiments for the motion uncertainty effect in a synthetic dataset using the expected kernel and the expected sub-map uncertainty propagation schemes. The experiment is repeated five times by increasing the robot motion noise covariance, i.e. $\boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_5$. Note that all mapping techniques are incremental.

Model selection:		
Technique	Cov. func.	Warping func.
GPOM	Matérn ($\nu = 5/2$)	n/a
WGPOM	SE (ARD)	tanh ($\ell = 2$)
Compared mapping techniques:		
Technique	Uncertainty propagation	Abbreviation
GPOM	EK	GEK
GPOM	ESM	GESM
WGPOM	EK	WEK
WGPOM	ESM	WESM
Numerical integration:		
Uncertainty propagation	Technique	Samples
EK	Gauss-Hermite	9
ESM	Monte-Carlo	10
Training and test (query) points size:		
Parameter	Symbol	Value
Tot. training points	n_t	87833
Ave. training points	$\bar{n}_t$	829
Test points	n_q	6561
Robot motion model noise covariances:		
$\boldsymbol{Q}_1 = \text{diag}(0.05\,\text{m}, 0.05\,\text{m}, 0.25\,\text{rad})^2$, $\boldsymbol{Q}_2 = \text{diag}(0.1\,\text{m}, 0.1\,\text{m}, 0.5\,\text{rad})^2$		
$\boldsymbol{Q}_3 = \text{diag}(0.15\,\text{m}, 0.15\,\text{m}, 0.75\,\text{rad})^2$, $\boldsymbol{Q}_4 = \text{diag}(0.2\,\text{m}, 0.2\,\text{m}, 1.00\,\text{rad})^2$		
$\boldsymbol{Q}_5 = \text{diag}(0.3\,\text{m}, 0.3\,\text{m}, 2.00\,\text{rad})^2$		

6.3.1 First Experiment: Motion Uncertainty Effect

The map of the environment, the robot trajectory, the observations collected at each pose using a simulated range-finder sensor, together with the evolution of the robot pose uncertainty due to its motion noise, are illustrated in Figure 6.4. The uncertainty ellipsoids show the worst-case covariance of the robot position, and there is no uncertainty reduction along the path by closing loops.

Details from the model selection, compared techniques, and conducted experiments are collected in Table 6.1. We use Matérn ($\nu = 5/2$) covariance function for GPOM as its performance has been shown fitting in Chapter 4. For WGPOM, we use SE covariance function with Automatic Relevance Determination (ARD) (Neal 1996). Since the observations do not cover the entire map, we

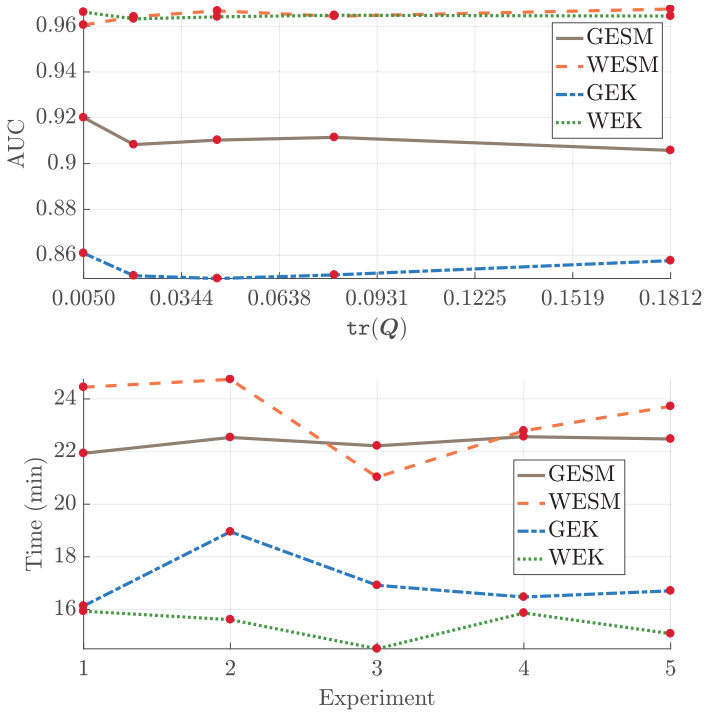


FIGURE 6.5: The AUC and runtime for incremental GPOM and WGPOM using the synthetic dataset and proposed uncertainty propagation methods. The runtimes are in minutes. All maps are computed using 0.5 m resolution and a kd-tree data structure for the storage and nearest neighbor queries. As it is expected, generally, by propagating pose uncertainties into the map inference, the map quality degrades (GESM and GEK). However, applying WGPs can alleviate this effect and produce maps with improved accuracies (WESM and WEK).

use tanh with $\ell = 2$ as the warping function to improve the extrapolation ability of GPs (Figure 6.3). The experiment for each mapping technique using EK and ESM uncertainty propagation is repeated by increasing the robot motion noise covariance in five steps. In Figure 6.5, the map accuracy and runtime comparisons of all methods using AUC are shown. In the bottom plot, the runtime for the ESM is higher than the EK for both mapping techniques. From the top plot, we can see that applying WGPs improves the map quality regardless of the uncertainty propagation choice. The expected kernel is computed using Gauss-Hermite quadrature with 9 sample points, and the expected sub-map computations are performed using Monte-Carlo approximations with 10 samples. When increasing the number of samples from 9 to 18 we did not observe any improvement in the map accuracy.

Figure 6.6 illustrates the results of all combinations of the proposed techniques. GPOM and WGPOM by ignoring the robot pose uncertainty are shown in Figures 6.6a and 6.6d, respectively. WGPOM demonstrates a better discrimination performance between class labels in the absence

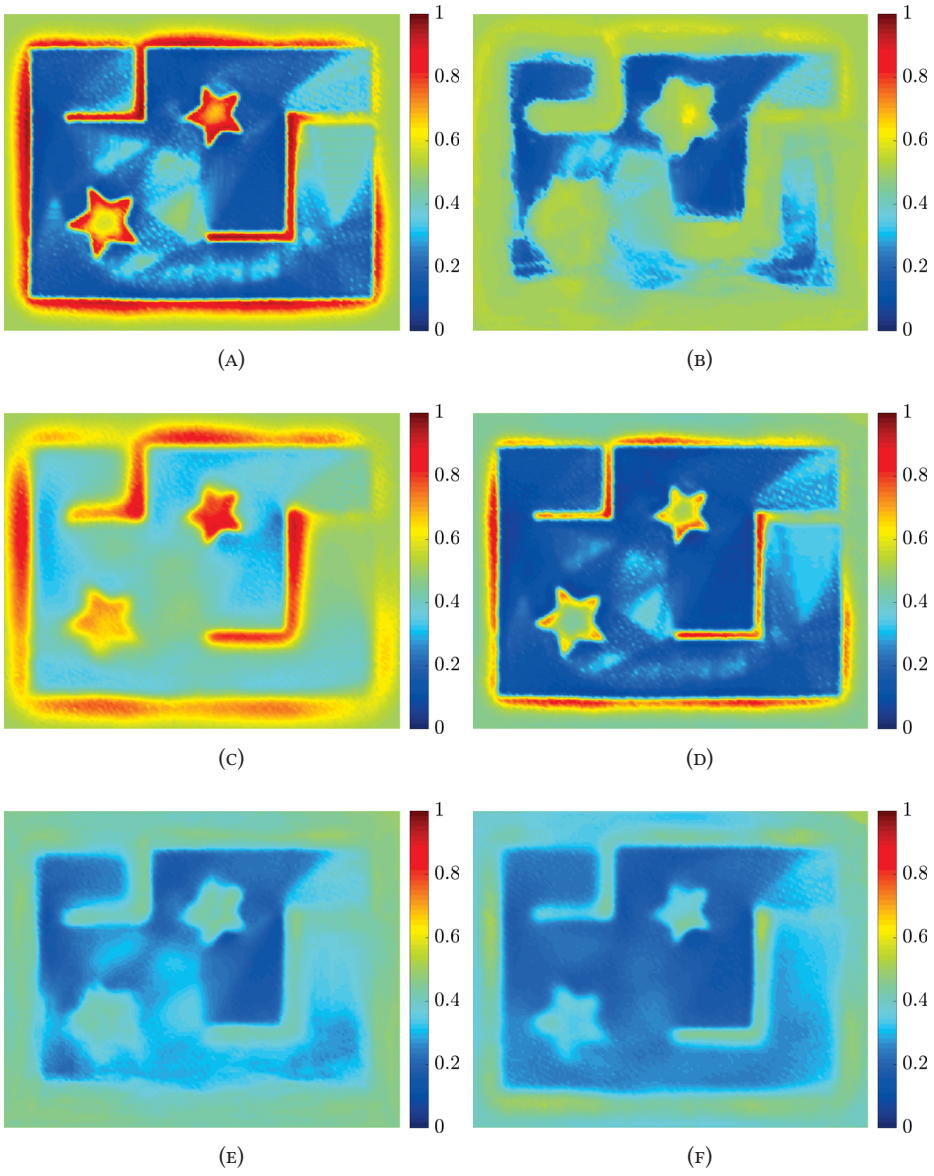


FIGURE 6.6: Illustrative examples of the occupancy map from the first experiment (for $\boldsymbol{Q}_3$). The top row corresponds to the GPOM, and the bottom row shows WGPOM results. The maps in (a) and (d) show GPOM and WGPOM results by ignoring the robot pose uncertainties. In (b) and (e) the robot pose uncertainty is incorporated using the expected sub-map method. In (c) and (f) the robot pose uncertainty is incorporated using the expected kernel method. The WGPOM-based maps can deal with input uncertainty better and provide maps with higher quality, shown in (e) and (f). All maps are computed using 0.5 m resolution.

TABLE 6.2: The AUC and runtime for incremental GPOM and WGPOM using the Intel dataset and proposed uncertainty propagation methods. The runtimes are in minutes. Maps are computed using 0.2 m resolution and a kd-tree data structure for the storage and nearest neighbor queries. The Sparse covariance function is used for all techniques and Polynomials of degree seven as the warping function. Total and average number of training points, and the number of test points are $n_t = 137979$, $\bar{n}_t = 186$, and $n_q = 40401$, respectively.

Method	No uncertainty		EK		ESM	
	AUC	Time	AUC	Time	AUC	Time
GPOM	0.8499	72	0.7323	532	0.7026	657
WGPOM	0.8463	125	0.7887	577	0.7436	766

of measurements. This effect can be seen in the middle of the smaller star in both maps. Note that further optimization of hyperparameters can lead to over-fitting instead of solving the discussed problem.

Even though ESM and EK try to achieve the same goal, they demonstrate different behaviors. Generally speaking, integration over the covariance function (EK) has a smoothing effect that sometimes can be desirable. For example, in all the presented maps the central part of the map due to the lack of observation is partially complete. As a result of this smoothing effect of the EK, this part is correctly classified to be closer to the free space class. However, the probabilities are closer to 0.5, and the robot cannot be completely confident about the status of the area. Alternatively, ESM leads to relatively more confident maps with smaller smoothing effects. The resultant maps are safer for navigation as the gap between occupied and unoccupied areas is classified as an unknown region. While this behavior can be an appealing property from a motion planning point of view, the occupied areas are faded; and we cannot see the structural shape of the environment accurately. Overall, by propagating more uncertainty into the map inference process, we expect less accuracy in the outcome and more realistic estimation of the belief. However, WGPs by modeling nonlinearity in the observation space improve the map quality.

6.3.2 Experimental Results

The experimental results of occupancy mapping using the Intel dataset are shown in Table 6.2 and Figure 6.7. In the absence of a complete ground truth map, the ground truth map for this dataset is generated using the estimated robot trajectory and rangefinder measurements. In this way, the ground truth map has the same orientation which makes the comparison convenient.

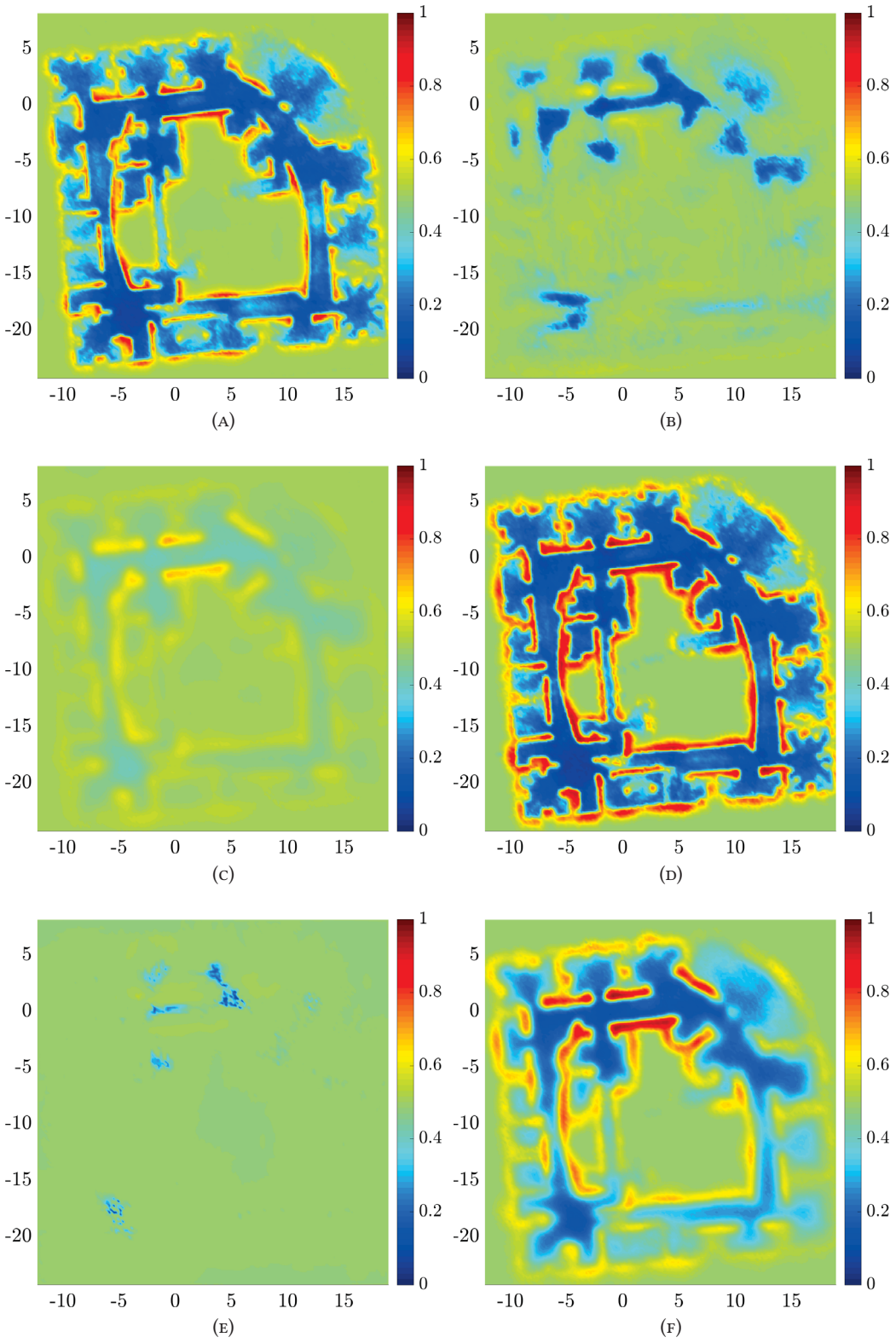


FIGURE 6.7: Occupancy mapping results using the Intel dataset. The top row corresponds to the GPOM, and the bottom row shows WGPOM results. The maps built by ignoring pose uncertainties are shown in (a) and (d) using GPOM and WGPOM, respectively. In (b) and (e) the robot pose uncertainty is incorporated using the expected sub-map method. In (c) and (f) the robot pose uncertainty is incorporated using the expected kernel method.

The covariance function used is an intrinsically sparse kernel (Melkumyan and Ramos 2009). The logic behind this choice is that the structural shape of the environment is complex, and it is cluttered with random people and typical office furniture; therefore, using a covariance function that correlates map points over a long range is not suitable. We use the Sparse covariance function for both GPOM and WGPOM and their corresponding uncertainty propagation experiments.

Table 6.2 shows the runtime and accuracy comparison of the mapping techniques. In this scenario, where the pose uncertainties are ignored, GPOM performs marginally better than WGPOM. This result, shown in Figures 6.7a and 6.7d, can be understood from the fact that WGPOM has covered more partially observed areas in the map. While this is desirable, in the absence of a complete ground truth map, it leads to a lower AUC. However, WGPOM maintains more accurate maps by incorporating the pose uncertainties which is the actual problem to be solved.

In this experiment, the uncertainty propagation using the ESM method leads to poor map qualities and the maps, shown in Figures 6.7b and 6.7e, are almost entirely faded due to the significant pose uncertainties resulting from the robot repeatedly traveling through the same areas. This fading effect could be partially mitigated by increasing the number of samples, but based on the runtimes in Table 6.2, it is not justifiable. Nevertheless, WGPOM using the EK demonstrates a better performance in comparison to all methods that incorporated pose uncertainties; moreover, it produces a map that is also usable for navigation tasks, i.e. obstacle avoidance, and path planning.

6.3.3 Discussion and Limitations

Uncertainty propagation through the developed methods can provide safer maps for robotic navigation. However, long-term uncertainty propagation leads to highly faded maps. If the uncertainty at each step is large and the robot cannot improve the localization confidence through loop-closures, this fading effect is more severe. Also, the uncertainty of the robot orientation has a large impact on the map quality. As we have seen in the presented results, in the expected submap approach, if the orientation uncertainty is significant, integration using a small number of samples leads to poor map accuracies. The expected kernel technique has an advantage from this perspective, as the unscented transform maps the measurement into the map space, and samples are related to the map spatial dimensions.

It is also demonstrated that the concept of accounting for possible nonlinearities in the observation space, here through the warping function, has desirable effects on the map quality. Our results showed that regardless of the uncertainty propagation technique, applying WGPs provide better maps than standard GPs. We reiterate that by ignoring the robot pose uncertainty, the map is a potentially incorrect representation of the environment.

Finding an appropriate warping function that is compatible with non-Gaussianity in the observation space can be time-consuming unless the model selection is performed in a more systematic way. In this work, we tried several functions and chose the best one. Moreover, since the exact inference is not possible, approximate methods may not always converge. Although the upper-bound time complexity of WGPOM is similar to that of GPOM, in practice for larger datasets the inference takes longer. Improving the computational efficiency of the proposed methods is an interesting direction to follow.

6.3.4 Computational Complexity

For both GPOM and WGPOM, the worst-case time complexity is cubic in the number of training data, $\mathcal{O}(\bar{n}_t^3)$. For ESM, the number of sub-map fusion into the global map scales linearly with the number of samples, n_s, and the sub-map fusion involves a nearest neighbor query for each test point resulting in $\mathcal{O}(\bar{n}_t^3 + n_s n_q \log n_q)$. In the case of two-dimensional mapping, using Gauss-Hermite quadrature, the time complexity of EK computation is quadratic in the number of sample points and together with sub-map fusion leads to $\mathcal{O}(\bar{n}_t^3 + n_s^2 + n_q \log n_q)$. In EK, applying the unscented transform to all training points involves a Cholesky factorization of the input covariance matrix which is ignored for the two-dimensional case.

6.4 Chapter Summary

In this chapter, we studied incremental GP occupancy mapping extensions through WGPs. Since occupancy maps have dense belief representations, the robot pose uncertainty is often ignored. We proposed two methods to incorporate robot pose uncertainty into the map inference, the expected kernel and the expected sub-map. While the expected kernel handles the input uncertainty within the GPs framework, the expected sub-map exploits the inherent property of

stationary covariance functions for map inference in the local frame with deterministic inputs. The proposed methods can also be useful if the belief representation is not dense (as opposed to occupancy mapping). Furthermore, the WGPOM technique can deal with the nonlinear behavior of measurements through a nonlinear transformation which improves the ability of GPs to learn complex structural shapes more accurately, especially, under uncertain inputs. In the following chapter, we study a more general class of motion planning using a sampling-based strategy, and we use the expected kernel method developed here for the prediction purpose.

Chapter 7

Sampling-based Incremental Information Gathering

In this chapter, we propose a sampling-based motion planning algorithm equipped with an information-theoretic convergence criterion for incremental informative motion planning. The proposed approach allows for a dense map representation and incorporates the full state uncertainty into the planning process. The problem is formulated as a maximization problem with a budget constraint. Our approach is built on rapidly-exploring information gathering algorithms and benefits from advantages of sampling-based optimal motion planning algorithms. We propose two information functions and their variants for fast and online computations. We prove an information-theoretic convergence for the entire exploration and information gathering mission based on the least upper bound of the average map entropy. The convergence criterion gives rise to a natural automatic stopping criterion for information-driven motion control. We demonstrate the performance of the proposed algorithms using three scenarios: comparison of the proposed information functions and sensor configuration selection, robotic exploration in unknown environments, and a wireless signal strength monitoring task in a lake from a publicly available dataset collected using an autonomous surface vehicle.

Exploration in unknown environments is a major challenge for an autonomous robot and has numerous present and emerging applications from search and rescue operations to space exploration programs. While exploring an unknown environment, the robot is often tasked to monitor

a quantity of interest through a cost or an information quality measure (Dhariwal et al. 2004; Singh et al. 2010; Marchant and Ramos 2012; Dunbabin and Marques 2012; Lan and Schwager 2013; Yu et al. 2015). Robotic exploration algorithms usually rely on geometric frontiers (Yamauchi 1997; Ström et al. 2015) or visual targets (Kim and Eustice 2015) as goals to solve the planning problem using geometric/information gain-based *greedy* action selection or planning for a limited horizon. The main drawback of such techniques is that a set of targets constrains the planner search space to the paths that start from the current robot pose to targets. In contrast, a more general class of robotic navigation problem known as robotic information gathering (Singh et al. 2009; Binney and Sukhatme 2012; Binney et al. 2013), and in particular the Rapidly-exploring Information Gathering (RIG) (Hollinger and Sukhatme 2013, 2014) concept exploits a sampling-based planning strategy to calculate the cost and information gain while searching for traversable paths. This approach differs from classic path planning problems as there is no goal to be found. Therefore, a solely information-driven robot control can be formulated. Another important advantage of RIG methods is their *multi-horizon planning* nature through representing the environment by an incrementally built graph that considers both the information gain and cost.

Rapidly-exploring information gathering algorithms are suitable for non-myopic robotic exploration techniques. However, the developed methods are not online, the information function calculation is often a bottleneck, and they do not offer an automated convergence criterion, i.e. they are *anytime* [1]. To be able to employ RIG for online navigation tasks, we propose an Incrementally-exploring Information Gathering (IIG) algorithm built on the RIG. In particular, the following developments are performed:

- We allow the full state including the robot pose and a dense map to be partially observable.
- We develop a convergence criterion based on relative information contribution. This convergence criterion is a necessary step for incremental planning as the robot needs to execute planned actions autonomously.
- We propose two information functions (five including their variants) that can approximate the information gain for online applications. The algorithmic implementation of the proposed functions is also provided.

[1] Being anytime is typically a good feature, but we are interested in knowing when the algorithm converges.

– We develop a heuristic algorithm to extract the most informative trajectories.

– We prove an information-theoretic automatic stopping criterion for the entire mission based on the least upper bound of the average map entropy.

– We provide results in batch and incremental experiments as well as in publicly available robotic datasets and discuss potential applications of the developed algorithms.

This chapter is organized as follows. In the next section, we present the problem definition and make the difference between RIG and IIG algorithms clear. We explain RIG algorithms to establish the required basis. In Section 7.2, we present the novel IIG algorithm. In Section 7.3, we discuss the proposed information functions to approximate the information quality of nodes and trajectories. In Section 7.4, the problem of path extraction and selection is explained and a heuristic algorithm is proposed to solve the problem. Section 7.5 presents theoretical analyses of the proposed algorithms including proving an information-theoretic automatic stopping criterion for exploration and information gathering missions. In Section 7.6, we present results from the comparison of the proposed information functions, exploration experiments including comparison with relevant techniques in the literature, a lake monitoring scenario, and discuss the limitations and possible extension of this work. Finally, Section 7.7 concludes the chapter.

7.1 Problem Statement and Preliminaries

The problem of robotic information gathering is formulated as a maximization problem subject to finite resources, i.e. a budget b. In Hollinger and Sukhatme (2014), this problem is defined as follows.

Problem 7.1 (Informative motion planning)**.** *Let $\mathcal{A}$ be the space of all possible trajectories and $f_I(\mathcal{P})$ be a function that quantifies the information quality along a trajectory $\mathcal{P}$. Let $f_c(\mathcal{P})$ be a function that returns the cost associated with trajectory $\mathcal{P}$. Given the available budget b, the problem can be formulated as following.*

$$\mathcal{P}^* = \underset{\mathcal{P}\in\mathcal{A}}{\arg\max}\ f_I(\mathcal{P}) \quad s.t.\ f_c(\mathcal{P}) \leq b \tag{7.1}$$

Now we express the assumptions in RIG algorithms.

Assumption 7.2. *The cost function $f_c(\mathcal{P})$ is strictly positive, monotonically increasing, bounded, and additive such as distance and energy.*

Remark 7.3. *The information function $f_I(\mathcal{P})$ can be modular, time-varying modular, or submodular.*

The information function assumption follows from Hollinger and Sukhatme (2014), even though we focus our attention on the submodular class of information functions as the information gathered at any future time during navigation depends on prior robot trajectories. Another reason to consider submodular information functions is to avoid information "double-counting". This choice allows us to develop an information-theoretic convergence criterion for RIG/IIG as the amount of available information remains bounded. The following assumptions are directly from Hollinger and Sukhatme (2014) which in turn are equivalent or adapted from Bry and Roy (2011); Karaman and Frazzoli (2011). The `Steer` function used in Assumption 7.4 extends nodes towards newly sampled points.

Assumption 7.4. *Let $\boldsymbol{x}_a, \boldsymbol{x}_b$, and $\boldsymbol{x}_c$ be three points within radius Δ of each other. Let the trajectory e_1 be generated by $\texttt{Steer}(\boldsymbol{x}_a, \boldsymbol{x}_c, \Delta)$, e_2 be generated by $\texttt{Steer}(\boldsymbol{x}_a, \boldsymbol{x}_b, \Delta)$, and e_3 be generated by $\texttt{Steer}(\boldsymbol{x}_b, \boldsymbol{x}_c, \Delta)$. If $\boldsymbol{x}_b \in e_1$, then the concatenated trajectory $e_2 + e_3$ must be equal to e_1 and have equal cost and information.*

This assumption is required as in the limit drawn samples are infinitely close together, and the `Steer` function, cost, and information need to be consistent for any intermediate point.

Assumption 7.5. *There exists a constant $r \in \mathbb{R}_{>0}$ such that for any point $\boldsymbol{x}_a \in \mathcal{X}_f$ there exists an $\boldsymbol{x}_b \in \mathcal{X}_f$, such that 1) the ball of radius r centered at $\boldsymbol{x}_a$ lies inside $\mathcal{X}_f$ and 2) $\boldsymbol{x}_a$ lies inside the ball of radius r centered at $\boldsymbol{x}_b$.*

This assumption ensures that there is enough free space near any point for extension of the graph. Violation of this assumption in practice can lead to failure of the algorithm to find a path.

Assumption 7.6 (Uniform sampling)**.** [2] *Points returned by the sampling function `sample` are independent and identically distributed (i.i.d.) and drawn from a uniform distribution.*

[2] Results extend naturally to any absolutely continuous distribution with density bounded away from zero on workspace $\mathcal{X}$ (Karaman and Frazzoli 2011).

7.1.1 Incremental Informative Motion Planning

Now we define the problem of incremental informative motion planning as follows.

Problem 7.7 (Incremental informative motion planning)**.** *Let $s_{0:t_s} \in \mathcal{S}$ be the current estimate of the state up to time t_s. Let $\mathcal{A}_t$ be the space of all possible trajectories at time t and $f_I(\mathcal{P}_t)$ be a function that quantifies the information quality along a trajectory $\mathcal{P}_t$. Let $f_c(\mathcal{P}_t)$ be a function that returns the cost associated with trajectory $\mathcal{P}_t$. Given the available budget b_t, the problem can be formulated as following:*

$$\mathcal{P}_t^* = \underset{\mathcal{P}_t \in \mathcal{A}_t}{\arg\max}\ f_I(\mathcal{P}_t) \quad \forall\ t > t_s \quad s.t.\ f_c(\mathcal{P}_t) \leq b_t \text{ and } S = s_{0:t_s} \tag{7.2}$$

Remark 7.8. *The state S can include the representation of the environment (map), the robot trajectory, and possibly any other variables defined in the state vector. In general, the information function $f_I(\mathcal{P}_t)$ is responsible for incorporating the state uncertainty in the information gain calculations.*

Remark 7.9. *In practice we solve the problem incrementally and use a planning horizon $T > t_s$ that in the limit goes to ∞.*

The main difference between Problem 7.1 and Problem 7.7 is that in the latter, the robot does not have the full knowledge of the environment *a priori*. Therefore Problem 7.7 is not only the problem of information gathering but also *planning for estimation* as the robot needs to infer the map (and in general its pose in the SLAM problem) sequentially. Note that we do not impose any assumptions on the observability of the robot pose and the map, therefore, they can be partially observation as the case in POMDPs.

The problems mentioned earlier are both in their offline (nonadaptive) and online (adaptive) forms NP-hard (Singh et al. 2009). We build our proposed incremental information gathering algorithm on top of RIG to solve the interesting problem of autonomous robotic exploration in unknown environments. Furthermore, since the ultimate goal is online applications, we only consider the RIG-tree variant to be extended for sequential planning. This conclusion stems from extensive comparisons of RIG variants provided in Hollinger and Sukhatme (2014). However, we acknowledge that the RIG-graph is an interesting case to consider as under a *partial ordering* assumption it is *asymptotically optimal*.

Algorithm 8 `RIG-tree()`

Require: Step size Δ, budget b, free space $\mathcal{X}_f$, Environment $\mathcal{M}$, start configuration $\boldsymbol{x}_{start}$, near radius r;

// Initialize cost, information, starting node, node list, edge list, and tree
$I_{init} \leftarrow$ `Information`$([\,], \boldsymbol{x}_{start}, \mathcal{M}), C_{init} \leftarrow 0, n \leftarrow \langle \boldsymbol{x}_{start}, C_{init}, I_{init} \rangle$
$\mathcal{V} \leftarrow \{n\}, \mathcal{V}_{closed} \leftarrow \varnothing, \mathcal{E} \leftarrow \varnothing$
while not terminated **do**
    // Sample configuration space of vehicle and find nearest node
    $\boldsymbol{x}_{sample} \leftarrow$ `Sample`$(\mathcal{X}_f)$
    $\boldsymbol{x}_{nearest} \leftarrow$ `Nearest`$(\boldsymbol{x}_{sample}, \mathcal{V} \backslash \mathcal{V}_{closed})$
    $\boldsymbol{x}_{feasible} \leftarrow$ `Steer`$(\boldsymbol{x}_{nearest}, \boldsymbol{x}_{sample}, \Delta)$
    // Find near points to be extended
    $\mathcal{V}_{near} \leftarrow$ `Near`$(\boldsymbol{x}_{feasible}, \mathcal{V} \backslash \mathcal{V}_{closed}, r)$
    for all $n_{near} \in \mathcal{V}_{near}$ **do**
        // Extend towards new point
        $\boldsymbol{x}_{new} \leftarrow$ `Steer`$(\boldsymbol{x}_{near}, \boldsymbol{x}_{feasible}, \Delta)$
        if `NoCollision`$(\boldsymbol{x}_{near}, \boldsymbol{x}_{new}, \mathcal{X}_f)$ **then**
            // Calculate new information and cost
            $I_{new} \leftarrow$ `Information`$(I_{near}, \boldsymbol{x}_{new}, \mathcal{M}), c(\boldsymbol{x}_{new}) \leftarrow$ `Cost`$(\boldsymbol{x}_{near}, \boldsymbol{x}_{new})$
            $C_{new} \leftarrow C_{near} + c(\boldsymbol{x}_{new}), n_{new} \leftarrow \langle \boldsymbol{x}_{new}, C_{new}, I_{new} \rangle$
            if `Prune`(n_{new}) **then**
                delete n_{new}
            else
                // Add edges and nodes
                $\mathcal{E} \leftarrow \cup\{(n_{near}, n_{new})\}, \mathcal{V} \leftarrow \cup\{n_{new}\}$
                // Add to closed list if budget exceeded
                if $C_{new} > b$ **then**
                    $\mathcal{V}_{closed} \leftarrow \mathcal{V}_{closed} \cup \{n_{new}\}$
                end if
            end if
        end if
    end for
end while
return $\mathcal{T} = (\mathcal{V}, \mathcal{E})$

7.1.2 RIG Algorithms

The sampling-based RIG algorithms find a trajectory that maximizes an information quality metric with respect to a pre-specified budget constraint (Hollinger and Sukhatme 2014). The RIG is based on RRT*, RRG, and PRM* (Karaman and Frazzoli 2011) and borrows the notion of informative path planning from the branch and bound optimization (Binney and Sukhatme 2012). Algorithm 8 shows the RIG-tree algorithm. The functions that are used in the algorithm are explained as follows.

`Cost` – The cost function assigns a strictly positive cost to a collision-free path between two points from the free space $\mathcal{X}_f$.

`Information` – This function quantifies the information quality of a collision-free path between two points from the free space $\mathcal{X}_f$.

`Sample` – This function returns i.i.d. samples from $\mathcal{X}_f$.

`Nearest` – Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} \subset \mathcal{X}_f$, and a query point $\boldsymbol{x} \in \mathcal{X}_f$, this function returns a vertex $v \in \mathcal{V}$ that has the "closest" distance to the query point [3].

`Steer` – This function extends nodes towards newly sampled points and constrains the motion of the robot [4].

`Near` – Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} \subset \mathcal{X}_f$, a query point $\boldsymbol{x} \in \mathcal{X}_f$, and a positive real number $r \in \mathbb{R}_{>0}$, this function returns a set of vertices $\mathcal{V}_{near} \subseteq \mathcal{V}$ that are contained in a ball of radius r centered at $\boldsymbol{x}$.

`NoCollision` – Given two points $\boldsymbol{x}_a, \boldsymbol{x}_b \in \mathcal{X}_f$, this functions returns **true** if the line segment between $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ is collision-free and **false** otherwise.

`Prune` – This function implements a pruning strategy to remove nodes that are not "promising". This can be achieved through defining a *partial ordering* for co-located nodes.

In line 2-3 the algorithm initializes the starting node of the graph (tree). In line 6-8, a sample point from workspace $\mathcal{X}$ is drawn and is converted to a feasible point, from its nearest neighbor in the graph. Line 10 extracts all nodes from the graph that are within radius r of the feasible point. These nodes are candidates for extending the graph and each node is converted to a new node using the `Steer` function in line 13. In line 14-17, if there exists a collision-free path between the candidate node and the new node, the information gain and cost of the new node are evaluated. In line 18-25, if the new node does not satisfy a partial ordering condition is pruned. Otherwise, it is added to the graph. Furthermore, the algorithm checks for the budget constraint violation. The output is a graph that contains a subset of traversable paths with maximum information gain.

[3] Here we use Euclidean distance.
[4] Through this function, it is possible to make the planner kinodynamic.

7.2 IIG: Incrementally-exploring Information Gathering

In this section, we present the IIG algorithm which is essentially RIG with an information-theoretic convergence condition. The algorithmic implementation of IIG is shown in Algorithm 9. We employ IIG to solve the robotic exploration problem with the partially observable state. Both RIG and IIG, through incremental sampling, search the space of possible trajectories to find the maximally informative path; however, due to the automatic convergence of the IIG it is possible to run the algorithm online without the full knowledge of the state, i.e. the map and robot poses.

We introduce the Relative Information Contribution (RIC) criterion to detect the convergence of the search. The motivation behind this definition is that the number of nodes constantly increases unless an aggressive pruning strategy is used. However, an aggressive pruning strategy leads to potentially pruning nodes that can be part of optimal solutions [5]. Even though the algorithm continues to add nodes, it is possible to evaluate the contribution of each added node in the relative information sense. In other words, adding nodes does not affect the convergence of the algorithm, but the amount of information the algorithm can collect by continuing the search. We define the RIC of a node as follows.

Definition 7.10 (Relative Information Contribution)**.** *In Algorithm 9, let $\mathbf{x}_{new} \in \mathcal{X}_f$ be a reachable point through a neighboring node $n_{near} \in \mathcal{V}$ returned by the function* `Near`*(). Let I_{new} and I_{near} be the information values of their corresponding nodes returned by the function* `Information`*(). The relative information contribution of node n_{new} is defined as*

$$RIC \triangleq \frac{I_{new}}{I_{near}} - 1 \tag{7.3}$$

Equation (7.3) is conceptually important as it defines the amount of information gain relative to a neighboring point in the IIG graph. In practice, the number of samples it takes before the algorithm finds a new node becomes important. Thus we define penalized relative information contribution that is computed in line 24.

Definition 7.11 (Penalized Relative Information Contribution)**.** *Let RIC be the relative information contribution computed using Equation* (7.3)*. Let n_{sample} be the number of samples it takes to*

[5] Note that more than one optimal trajectory at each time can exist, e.g. when the robot needs to explore two equally important directions.

Algorithm 9 `IIG-tree()`

Require: Step size Δ, budget b, free space $\mathcal{X}_f$, Environment $\mathcal{M}$, start configuration $\boldsymbol{x}_{start}$, near radius r, relative information contribution threshold δ_{RIC}, averaging window size n_{RIC};

// Initialize cost, information, starting node, node list, edge list, and tree
$I_{init} \leftarrow \texttt{Information}([\,], \boldsymbol{x}_{start}, \mathcal{M}), C_{init} \leftarrow 0, n \leftarrow \langle \boldsymbol{x}_{start}, C_{init}, I_{init} \rangle$
$\mathcal{V} \leftarrow \{n\}, \mathcal{V}_{closed} \leftarrow \varnothing, \mathcal{E} \leftarrow \varnothing$
$n_{sample} \leftarrow 0$ // Number of samples
$I_{RIC} \leftarrow \varnothing$ // Relative information contribution
while $\texttt{AverageRIC}(I_{RIC}, n_{RIC}) > \delta_{RIC}$ **do**
    // Sample configuration space of vehicle and find nearest node
    $\boldsymbol{x}_{sample} \leftarrow \texttt{Sample}(\mathcal{X}_f)$
    $n_{sample} \leftarrow n_{sample} + 1$
    $\boldsymbol{x}_{nearest} \leftarrow \texttt{Nearest}(\boldsymbol{x}_{sample}, \mathcal{V} \backslash \mathcal{V}_{closed})$
    $\boldsymbol{x}_{feasible} \leftarrow \texttt{Steer}(\boldsymbol{x}_{nearest}, \boldsymbol{x}_{sample}, \Delta)$
    // Find near points to be extended
    $\mathcal{V}_{near} \leftarrow \texttt{Near}(\boldsymbol{x}_{feasible}, \mathcal{V} \backslash \mathcal{V}_{closed}, r)$
    for all $n_{near} \in \mathcal{V}_{near}$ **do**
        // Extend towards new point
        $\boldsymbol{x}_{new} \leftarrow \texttt{Steer}(\boldsymbol{x}_{near}, \boldsymbol{x}_{feasible}, \Delta)$
        if $\texttt{NoCollision}(\boldsymbol{x}_{near}, \boldsymbol{x}_{new}, \mathcal{X}_f)$ **then**
            // Calculate new information and cost
            $I_{new} \leftarrow \texttt{Information}(I_{near}, \boldsymbol{x}_{new}, \mathcal{M}), c(\boldsymbol{x}_{new}) \leftarrow \texttt{Cost}(\boldsymbol{x}_{near}, \boldsymbol{x}_{new})$
            $C_{new} \leftarrow C_{near} + c(\boldsymbol{x}_{new}), n_{new} \leftarrow \langle \boldsymbol{x}_{new}, C_{new}, I_{new} \rangle$
            if $\texttt{Prune}(n_{new})$ **then**
                delete n_{new}
            else
                $I_{RIC} \leftarrow \texttt{append}(I_{RIC}, (\frac{I_{new}}{I_{near}} - 1)/n_{sample})$ // Equation (7.4)
                $n_{sample} \leftarrow 0$ // Reset sample counter
                // Add edges and nodes
                $\mathcal{E} \leftarrow \cup\{(n_{near}, n_{new})\}, \mathcal{V} \leftarrow \cup\{n_{new}\}$
                // Add to closed list if budget exceeded
                if $C_{new} > b$ **then**
                    $\mathcal{V}_{closed} \leftarrow \mathcal{V}_{closed} \cup \{n_{new}\}$
                end if
            end if
        end if
    end for
end while
return $\mathcal{T} = (\mathcal{V}, \mathcal{E})$

find the node n_{new}. The penalized relative information contribution is defined as

$$I_{RIC} \triangleq \frac{RIC}{n_{sample}} \tag{7.4}$$

An appealing property of I_{RIC} is that it is non-dimensional and it does not depend on the actual calculation/approximation of the information values. In practice, as long as the information function satisfies the RIG/IIG requirements, using the following condition, the IIG algorithm converges. Let δ_{RIC} be a threshold that is used to detect the convergence of the algorithm. Through averaging I_{RIC} values over a window of size n_{RIC}, we ensure that continuing the search will not add any significant amount of information to the IIG graph. In Algorithm 9, this condition is shown in line 6 by function `AverageRIC`.

Remark 7.12. *In Algorithm 9, δ_{RIC} sets the planning horizon from the information gathering point of view. Through using smaller values of δ_{RIC} the planner can reach further points in both spatial and belief space. In other words, if $\delta_{RIC} \to 0$, then $T \to \infty$.*

7.3 Information Functions Algorithms

We propose two algorithms together with their variations to approximate the information gain at any sampled point from the free workspace. The information function in RIG/IIG algorithms often causes a bottleneck and computationally dominates the other parts. Therefore, even for offline calculations, it is important to have access to functions that, concurrently, are computationally tractable and can capture the essence of information gathering. We emphasize that the information functions are directly related to the employed sensors. However, once the model is provided and incorporated into the estimation/prediction algorithms, the information-theoretic aspects of the provided algorithms remain the same.

The information functions that are proposed are different in nature. First, we discuss MI-based information functions whose calculations explicitly depend on the sensor model. We provide two variants of Algorithm 7 that are based on forward and inverse sensor model map predictions. We also present an algorithm to approximate MI upper bound which reveals the maximum achievable information gain.

Then, we exploit the property of GPs to approximate the information gain. In Equation (3.33), repeated here for convenience

$$\sigma = \mathbb{V}[f_*] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x}_*)^T [\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \boldsymbol{I}_n]^{-1} \boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x}_*)$$

the variance calculation does not depend on the target vector (measurements). In this case, the information gain can be calculated using only map prior and posterior variances which removes the need for relying on a specific sensor model and calculating the expectation over future measurements. Once we established Gaussian Processes Variance Reduction (GPVR) algorithm, we use the expected kernel notion to propagate pose uncertainty into the covariance function resulting in Uncertain Gaussian Processes Variance Reduction (UGPVR) algorithm. In particular, GPVR-based information functions are interesting for the following reasons:

- Unlike MI-based (direct information gain calculation), they are non-parametric.
- GPVR-based information functions provide a systematic way to incorporate pose uncertainty into information gathering frameworks.
- In the case of incomplete knowledge about the quantity of interest in an unknown environment, they allow for *active learning* [6].

7.3.1 Mutual Information

To be able to calculate MI using map prediction we need to update the map after every measurement prediction. It is possible to perform map prediction using a forward or inverse sensor model. Typically using an inverse sensor model results in simpler calculations. Nevertheless, we provide two algorithms that use both types of map predictions. First, we define necessary parameters required in the proposed algorithms.

Definition 7.13 (Map saturation probability)**.** *The probability that the robot is completely confident about the occupancy status of a point is defined as* p_{sat}*.*

Definition 7.14 (Map saturation entropy)**.** *The entropy of a point from a map whose occupancy probability is* p_{sat}*, is defined as* $h_{sat} \triangleq H(p_{sat})$*.*

The defined parameters are relevant since they prevent the exhaustive search for information in "less important" areas. The MI-based information function using a forward sensor model implementation is shown in Algorithm 10. In line 4, the predicted measurement for beam k is

[6] Although this is one of the most interesting aspects of GPVR-based information functions, it is beyond the scope of this thesis and we leave it as a possible extension of this work.

Algorithm 10 `InformationMI()`

Require: Robot pose or desired location, current map estimate m, sensor model, saturation entropy h_{sat};

1: $\bar{m} \leftarrow m$
2: $I \leftarrow 0$
3: **for** all k **do**
4: Compute $\hat{z}_{t+1}^{[k]}$ and $\mathcal{I}_{t+1}^{[k]}$ using ray casting in m
5: // Current map entropy along beam k
6: $h = -\sum_{i \in \mathcal{I}_{t+1}^{[k]}} [\bar{m}^{[i]} \log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]}) \log(1 - \bar{m}^{[i]})]$
7: $I = I + h$
8: // Calculate map conditional entropy along beam k
9: **for** $i \in \mathcal{I}_{t+1}^{[k]}$ **do**
10: $h_i = -[\bar{m}^{[i]} \log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]}) \log(1 - \bar{m}^{[i]})]$
11: **if** $h_i < h_{sat}$ **then**
12: $I = I - h_i$
13: **continue**
14: **end if**
15: $\bar{h} \leftarrow 0$
16: **for** all $z \leq \hat{z}_{t+1}^{[k]}$ **do**
17: // Calculate marginal measurement probability p_z
18: $p_1 \leftarrow p(\hat{z}_{t+1}^{[k]} | M = 0)$
19: $p_2 \leftarrow 0$
20: **for** $j \in \mathcal{I}_{t+1}^{[k]}$ **do**
21: $p_1 \leftarrow p_1 (1 - m^{[j]})$
22: $p_2 \leftarrow p_2 + p(\hat{z}_{t+1}^{[k]} | M = m^{[j]}) m^{[j]} \prod_{l<j} (1 - m^{[l]})$
23: **end for**
24: $p_z \leftarrow p_1 + p_2$
25: // Map prediction at point i along beam k
26: $\bar{m}^{[i]} \leftarrow p_z^{-1} p(\hat{z}_{t+1}^{[k]} | M = m^{[i]}) m^{[i]} \prod_{l<i} (1 - m^{[l]})$
27: $\bar{h} \leftarrow \bar{h} + p_z [\bar{m}^{[i]} \log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]}) \log(1 - \bar{m}^{[i]})]$
28: **end for**
29: $I \leftarrow I + \bar{h} s_z^{-1}$
30: **end for**
31: **end for**
32: **return** I (total information gain), $\bar{m}$ (estimated map)

computed using ray casting in the current map estimate. In line 7, for each beam k, the algorithm adds the total map entropy along beam k to the information gain, I. In line 11-14, the algorithm skips any map point whose entropy surpasses the saturation entropy and deducts its entropy from I. In line 15-28, the map conditional entropy by integrating over future measurement is calculated and in line 29 it is subtracted from initial map entropy using an appropriate numerical

Algorithm 11 `InformationMI2()`

Require: Robot pose or desired location, current map estimate m, sensor model, saturation probability p_{sat}, saturation entropy h_{sat}, free point belief b_{free}, occupied point belief b_{occ};

$\bar{m} \leftarrow m$
$I \leftarrow 0$
for all k **do**
  Compute $\hat{z}_{t+1}^{[k]}$ and $\mathcal{I}_{t+1}^{[k]}$ using ray casting in m
  // Current map entropy along beam k
  $h = -\sum_{i \in \mathcal{I}_{t+1}^{[k]}} [\bar{m}^{[i]} \log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]}) \log(1 - \bar{m}^{[i]})]$
  $I = I + h$
  // Calculate map conditional entropy along beam k
  for $i \in \mathcal{I}_{t+1}^{[k]}$ **do**
    $h_i = -[\bar{m}^{[i]} \log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]}) \log(1 - \bar{m}^{[i]})]$
    if $h_i < h_{sat}$ **then**
      $I = I - h_i$
      continue
    end if
    $\bar{h} \leftarrow 0$
    for all $z \leq \hat{z}_{t+1}^{[k]}$ **do**
      // Calculate marginal measurement probability p_z
      $p_1 \leftarrow p(\hat{z}_{t+1}^{[k]} | M = 0)$
      $p_2 \leftarrow 0$
      for $j \in \mathcal{I}_{t+1}^{[k]}$ **do**
        $p_1 \leftarrow p_1 (1 - m^{[j]})$
        $p_2 \leftarrow p_2 + p(\hat{z}_{t+1}^{[k]} | M = m^{[j]}) m^{[j]} \prod_{l<j} (1 - m^{[l]})$
      end for
      $p_z \leftarrow p_1 + p_2$
      // Map prediction at point i along beam k using inverse sensor model
      if `isFree`$(\bar{m}^{[i]})$ **then**
        $\bar{m}^{[i]} = \max(p_{sat} - \epsilon, b_{free} * \bar{m}^{[i]})$
      else
        $\bar{m}^{[i]} = \min(1 - p_{sat} + \epsilon, b_{occ} * \bar{m}^{[i]})$
      end if
      $\bar{h} \leftarrow \bar{h} + p_z [\bar{m}^{[i]} \log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]}) \log(1 - \bar{m}^{[i]})]$
    end for
    $I \leftarrow I + \bar{h} s_z^{-1}$
  end for
end for
return I (total information gain), $\bar{m}$ (estimated map)

integration resolution, s_z^{-1}.

Algorithm 12 `InformationMIUB()`

Require: Robot pose or desired location, current map estimate m, sensor model, saturation entropy h_{sat};

1: $\bar{m} \leftarrow m$
2: $I_{UB} \leftarrow 0$
3: **for** all k **do**
4: Compute $\hat{z}_{t+1}^{[k]}$ and $\mathcal{I}_{t+1}^{[k]}$ using ray casting in m
5: // Current map entropy along beam k
6: $h = -\sum_{i \in \mathcal{I}_{t+1}^{[k]}} [\bar{m}^{[i]} \log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]}) \log(1 - \bar{m}^{[i]})]$
7: $I_{UB} = I_{UB} + h$
8: // Check for saturated points along beam k
9: **for** $i \in \mathcal{I}_{t+1}^{[k]}$ **do**
10: $h_i = -[\bar{m}^{[i]} \log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]}) \log(1 - \bar{m}^{[i]})]$
11: **if** $h_i < h_{sat}$ **then**
12: $I_{UB} = I_{UB} - h_i$
13: **continue**
14: **end if**
15: Update the map points $\bar{m}^{[i]}$ along beam k using forward or inverse sensor model
16: **end for**
17: **end for**
18: **return** I_{UB} (total information gain), $\bar{m}$ (estimated map)

In practice, the map prediction using a forward sensor model used in Algorithm 10 for multi-step predictions may lead to poor map quality and, consequently, inaccurate information gain. Therefore, we develop Algorithm 11 that employs an inverse sensor model technique for map prediction. Algorithm 11 performs well in practice for multi-step predictions. The main difference can be seen in line 26-30 where map prediction is performed using free point belief, b_{free}, and occupied point belief, b_{occ}. To avoid losing numerical computation accuracy, the predicted probabilities are clamped using $\epsilon > 0$ which is a small number relative to p_{sat}.

It is also interesting to calculate an upper bound for the information gain. Given the provided algorithms, it is trivial to calculate MI upper bound using the total amount of map entropy in the current perception field of the robot. It is faster to compute the upper bound as it only shows the uncertainty from the current map and it does not consider any gain from future measurements. However, in practice, it can be useful for fast and online predictions. More details regarding the difference between maximizing mutual information and entropy are discussed in Guestrin et al. (2005); Krause et al. (2008).

Lemma 7.15 (Information gain upper bound)**.** *For any location in the map, the information gain*

upper bound is given by the total map entropy calculated using map points in the perception field of the robot at the same location.

Proof. From Lemma 3.5 and Theorem 3.6; $0 \leq I(M;Z) = H(M) - H(M|Z) \leq H(M)$ which extends to any sub-map $\mathcal{M}_{sub}$ that is in the perception field of the robot. Note that MI beyond the perception field is zero as $\forall\ M \in \mathcal{M} \setminus \mathcal{M}_{sub}$, $M \perp Z$. □

Algorithm 12 shows the Mutual Information Upper Bound (MIUB) information function in which the integration over predicted measurement is omitted. However, to avoid information double-counting, it is still required to update the map estimate after each function call. Furthermore, the MIUB calculation can be integrated into Algorithm 10 and 11 with an insignificant computational load.

7.3.2 GP Variance Reduction

Variance reduction is the essence of information gathering. Since predictive variance calculation in Equation (3.33), does not depend on observations, we can come up with a non-parametric algorithm to estimate variance reduction throughout dense belief representation of the map. For the problem of informative path planning, a similar approach is used in Binney et al. (2013) where the reduction in the trace of the covariance function is considered as the objective function (A-optimality). Here, we are interested in approximating the mutual information through entropy reduction, i.e. using determinant of the covariance matrix (D-optimality) (Pukelsheim 2006). This is mainly to keep the proposed IIG framework agnostic about the choice of information functions. We treat each map point as a continuous random variable that is normally distributed. Therefore we use differential entropy formulation for mutual information approximation.

Lemma 7.16 (Differential entropy of a Gaussian random variable)**.** *Let $X \sim \mathcal{N}(\mu, \sigma^2)$. The differential entropy of X can be derived as follows.*

$$h(X) = \frac{1}{2}\log(2\pi e\sigma^2) \tag{7.5}$$

Proof. The proof follows from the definition of differential entropy and direct integration (Cover and Thomas 1991). □

Proposition 7.17. *Let $X_1, X_2, ..., X_n$ have a multivariate normal distribution with covariance matrix $\boldsymbol{K}$. The mutual information between X and observations Z can be approximated as*

$$\hat{I}(X;Z) = \sum_{i=1}^{n} \log(\sigma_{X_i}) - \sum_{i=1}^{n} \log(\sigma_{X_i|Z}) \tag{7.6}$$

where σ_{X_i} and $\sigma_{X_i|Z}$ are marginal variances for X_i before and after incorporating observations Z, i.e. prior and posterior marginal variances.

Proof. Using marginalization property of normal distribution, Lemma 3.1, for every X_i we have $\mathbb{V}[X_i] = \boldsymbol{K}^{[i,i]}$. From Lemma 7.16, the mutual information for X_i can be written as

$$\hat{I}^{[i]}(X_i;Z) \approx \log(\sigma_{X_i}) - \log(\sigma_{X_i|Z}) \tag{7.7}$$

and the total mutual information can be calculated as $\hat{I}(X;Z) = \sum_{i=1}^{n} \hat{I}^{[i]}(X_i;Z)$. □

Algorithm 13 shows the details of GPVR information function [7]. Based on the training points generated from predicted measurement $\boldsymbol{z}$, a sub-map from the current map estimate using nearest neighbor search is found, line 3-8. In line 9-13, GP predictive variances are computed using covariance function k with the same hyperparameters learned for the map inference. In line 14-17, using BCM fusion the predictive marginal posterior variance is calculated and the information gain is updated consequently. The Cholesky factorization is the most computationally expensive operation of the algorithm. However, it is possible to exploit a sparse covariance matrix such as 3.40 or use a cut-off distance for the covariance function [8] to speed up the algorithm.

7.3.3 Uncertain GP Variance Reduction

Thus far, the developed information functions do not incorporate uncertainties of other state variables that are jointly distributed with the map (such as the robot pose) in information gain calculation. Using the expected kernel notion, Subsection 6.1.2, we define the modified kernel $\tilde{k}$ as follows.

[7] The algorithm uses MATLAB-style operations for matrix inversion, Cholesky factorization, and dot product with matrix inputs.

[8] The semi-positive definiteness of the covariance matrix needs to be maintained.

Algorithm 13 `InformationGPVR()`

Require: Robot pose or desired location $\boldsymbol{p}$, current map estimate m, covariance function k, sensor noise σ_n^2;

$\bar{\boldsymbol{\sigma}} \leftarrow \boldsymbol{\sigma}$ // global map variance
$I \leftarrow 0$
for all k **do**
    Compute $\hat{\boldsymbol{z}}_{t+1}^{[k]}$ and $\mathcal{I}_{t+1}^{[k]}$ using ray casting in m
end for
$\mathcal{D} \leftarrow \texttt{TrainingData}(\boldsymbol{p}, \boldsymbol{z})$
// Find the global sub-map using nearest neighbor search
$\mathcal{M}_{\mathcal{D}} \leftarrow \texttt{Near}(\mathcal{D})$
$\boldsymbol{K} \leftarrow k(\boldsymbol{X}, \boldsymbol{X}), \boldsymbol{K}_* \leftarrow k(\boldsymbol{X}, \boldsymbol{X}_*)$ // $\boldsymbol{X} \in \mathcal{M}_{\mathcal{D}}$ and $\boldsymbol{X}_* \in \mathcal{D}$
// Calculate vector of diagonal variances for training/test points
$\boldsymbol{k}_{**} \leftarrow k(\boldsymbol{X}_*, \boldsymbol{X}_*, \text{'diag'})$
$\boldsymbol{L} \leftarrow \texttt{Cholesky}(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})$, $\boldsymbol{V} \leftarrow \boldsymbol{L} \backslash \boldsymbol{K}_*$
$\boldsymbol{v} \leftarrow \boldsymbol{k}_{**} - \texttt{dot}(\boldsymbol{V}, \boldsymbol{V})^T$ // dot product
for all $i \in \mathcal{M}_{\mathcal{D}}$ **do**
    $\bar{\boldsymbol{\sigma}}^{[i]} \leftarrow ((\boldsymbol{\sigma}^{[i]})^{-1} + (\boldsymbol{v}^{[i]})^{-1})^{-1}$ // BCM fusion
    $I \leftarrow I + \log(\boldsymbol{\sigma}^{[i]}) - \log(\bar{\boldsymbol{\sigma}}^{[i]})$
end for
return I (total information gain), $\bar{\boldsymbol{\sigma}}$ (estimated map variance)

Definition 7.18 (Modified kernel)**.** *Let $k(x, x_*)$ be a kernel and $X \in \mathcal{X}$ a random variable that is distributed according to a probability distribution function $p(x)$. The modified kernel is defined as its expectation with respect to $p(x)$, therefore we can write*

$$\tilde{k} = \mathbb{E}[k] = \int_{\mathcal{X}} k p(x) dx \tag{7.8}$$

Through replacing the kernel function in Algorithm 13 with the modified kernel we can propagate the robot pose uncertainty in the information gain calculation. Intuitively, under the presence of uncertainty in other state variables that are correlated with the map, the robot does not take greedy actions as the amount of available information calculated using the modified kernel is less than the original case. Therefore, the chosen actions are relatively more conservative. The integration in Equation 7.8 can be numerically approximated using Monte-Carlo or Gauss-Hermite quadrature techniques. In the case of a Gaussian assumption for the robot pose, Gauss-Hermite quadrature provides a better accuracy and efficiency trade-off and is preferred.

Algorithm 14 shows UGPVR information function. The difference with GPVR is that the input location is not deterministic, i.e. it is approximated as a normal distribution $\mathcal{N}(\boldsymbol{p}, \Sigma)$, and the

Algorithm 14 `InformationUGPVR()`

Require: Robot pose or desired location $\boldsymbol{p}$ with covariance Σ, current map estimate m, modified covariance function $\tilde{k}$, sensor noise σ_n^2;

$\bar{\sigma} \leftarrow \sigma$ // global map variance
$I \leftarrow 0$
for all k **do**
    Compute $\hat{z}_{t+1}^{[k]}$ and $\mathcal{I}_{t+1}^{[k]}$ using ray casting in m
end for
$\mathcal{D} \leftarrow \texttt{TrainingData}(\boldsymbol{p}, \boldsymbol{z})$
// Find the global sub-map using nearest neighbor search
$\mathcal{M}_{\mathcal{D}} \leftarrow \texttt{Near}(\mathcal{D})$
$\boldsymbol{K} \leftarrow \tilde{k}(\boldsymbol{X}, \boldsymbol{X}, \Sigma), \boldsymbol{K}_* \leftarrow \tilde{k}(\boldsymbol{X}, \boldsymbol{X}_*, \Sigma)$ // $\boldsymbol{X} \in \mathcal{M}_{\mathcal{D}}$ and $\boldsymbol{X}_* \in \mathcal{D}$
// Calculate vector of diagonal variances for training/test points
$\boldsymbol{k}_{**} \leftarrow \tilde{k}(\boldsymbol{X}_*, \boldsymbol{X}_*, \Sigma, \text{'diag'})$
$\boldsymbol{L} \leftarrow \texttt{Cholesky}(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I})$, $\boldsymbol{V} \leftarrow \boldsymbol{L} \backslash \boldsymbol{K}_*$
$\boldsymbol{v} \leftarrow \boldsymbol{k}_{**} - \texttt{dot}(\boldsymbol{V}, \boldsymbol{V})^T$ // dot product
for all $i \in \mathcal{M}_{\mathcal{D}}$ **do**
    $\bar{\sigma}^{[i]} \leftarrow ((\sigma^{[i]})^{-1} + (\boldsymbol{v}^{[i]})^{-1})^{-1}$ // BCM fusion
    $I \leftarrow I + \log(\sigma^{[i]}) - \log(\bar{\sigma}^{[i]})$
end for
return I (total information gain), $\bar{\sigma}$ (estimated map variance)

covariance function is replaced by its modified version. Given the initial pose belief, the pose uncertainty propagation on the IIG graph can be performed using the robot motion model, i.e. using Equations (3.41) and (3.42). Since we already calculate the information gain for each node, it suffices to consider the worst-case covariance prediction. This approach is computationally more efficient and does not rely on any specific SLAM *front-end* for loop-closure detection.

7.4 Path Extraction and Selection

The RIG/IIG-tree algorithm provides a tree (graph) that is already expanded in the state space using maximum information gathering policy. However, we need to extract the maximally informative path as the selected action. For robotic exploration scenarios, it is not possible to traverse all the available trajectories because after execution of one trajectory [9] the state belief is updated and previous predictions are obsolete. Therefore, one possible solution will be finding a path in the IIG tree that maximizes the information gain.

[9] With the assumption that the robot remains committed to the selected action.

Algorithm 15 `PathSelection()`

Require: RIG/IIG tree $\mathcal{T}$, maximum path length l_{max}, maximum similar nodes n_{sim};

1: // Find all leaves using depth first search
2: $\mathcal{V}_{leaves} = \texttt{DFSpreorder}(\mathcal{T})$
3: // Find all paths by starting from each leaf and following parent nodes
4: $\mathcal{P}_{all} = \texttt{Paths2root}(\mathcal{T}, \mathcal{V}_{leaves})$
5: $l_{min} = \texttt{ceil}(\kappa l_{max})$ // Minimum path length, $0 < \kappa < 1$
6: **for** all $\mathcal{P} \in \mathcal{P}_{all}$ **do**
7: **if** $\texttt{length}(\mathcal{P}) \leq l_{min}$ **then**
8: **Delete** $\mathcal{P}$
9: **end if**
10: **end for**
11: $l = \texttt{length}(\mathcal{P}_{all})$
12: $vote = \texttt{zeros}(l, 1)$
13: // Find longest independent paths
14: **for** all $i < l - 1$ **do**
15: **for** all $i + 1 < j < l$ **do**
16: $l_{min} = \min(\texttt{length}(\mathcal{P}_i), \texttt{length}(\mathcal{P}_j))$
17: // Find number of common nodes between path i and j
18: $n_{com} = \texttt{SimilarNodes}(\mathcal{P}_i, \mathcal{P}_j)$
19: **if** $n_{com}/l_{min} > n_{sim}$ **then**
20: **if** $\texttt{length}(\mathcal{P}_i) > \texttt{length}(\mathcal{P}_j)$ **then**
21: $vote^{[i]} = vote^{[i]} + 1$
22: $vote^{[j]} = vote^{[j]} - 1$
23: **else**
24: $vote^{[i]} = vote^{[i]} - 1$
25: $vote^{[j]} = vote^{[j]} + 1$
26: **end if**
27: **else** // Two independent paths
28: $vote^{[i]} = vote^{[i]} + 1$
29: $vote^{[j]} = vote^{[j]} + 1$
30: **end if**
31: **end for**
32: **end for**
33: // Find paths with maximum vote and select the maximally informative path
34: $\mathcal{P}_{max} = \texttt{MaxVotePath}(\mathcal{P}_{all}, vote)$
35: $\mathcal{P}_I = \texttt{MaxInformativePath}(\mathcal{P}_{max})$
36: **return** $\mathcal{P}_I$

We provide a heuristic algorithm based on a voting method. Algorithm 15 shows the implementation of the proposed method. The algorithm first finds all possible paths using a preorder *depth first search*, function `DFSpreorder`, and then removes paths that are shorter than a minimum length (using parameter $0 < \kappa < 1$), line 2-10. Then each path is compared with others using the following strategy. If two paths have more than a specified number nodes in common, then we

penalize the shorter path by a negative vote and encourage the longer path by a positive vote. However, if two paths do not have many common nodes, then they are considered as two independent paths, and they receive positive votes, line 11-32. The function `SimilarNodes` returns the number of overlapping nodes between two paths. In line 34, the function `MaxVotePath` returns all paths that have the maximum number of votes. There is usually more than one path with the maximum vote, therefore, in line 35, the function `MaxInformativePath` selects the path that overall has the maximum information gain.

7.5 Information-theoretic Robotic Exploration

In this section, we present the information-theoretic basis for applying the IIG-tree algorithm to solve the autonomous robotic exploration problem. Since the developed algorithm does not rely on geometric features (frontiers) for map exploration, an alternative criterion is required for mission termination. We use the entropy independence bound theorem to leverage such a criterion.

Theorem 7.19 (Independence bound on entropy)**.** *Let $X_1, X_2, ..., X_n$ be drawn according to the joint distribution $p(x_1, x_2, ..., x_n)$. Then*

$$H(X_1, X_2, ..., X_n) \leq \sum_{i=1}^{n} H(X_i) \tag{7.9}$$

with equality if and only if the X_i are Independent.

Proof. The proof follows directly from Theorem 3.2 and 3.6. □

This theorem states that the joint entropy is always smaller than the sum of entropies independently, and both sides are equal if and only if the random variables are independent. We start from this inequality and prove that for robotic information gathering or map exploration the least upper bound of the average map entropy that is calculated by assuming independence between map points can be a threshold for mission termination.

Theorem 7.20 (The least upper bound of the average map entropy)**.** *Let $n \in \mathbb{N}$ be the number of map points. In the limit, for a completely explored occupancy map, the least upper bound of the average map entropy is given by $H(p_{sat})$.*

Proof. From Theorem 7.19 and through multiplying each side of the inequality by $\frac{1}{n}$, we can write the average map entropy as

$$\frac{1}{n}H(M) < \frac{1}{n}\sum_{i=1}^{n} H(M = m^{[i]}) \tag{7.10}$$

by taking the limit as $p(m) \to p_{sat}$, then

$$\lim_{p(m)\to p_{sat}} \frac{1}{n}H(M) < \lim_{p(m)\to p_{sat}} \frac{1}{n}\sum_{i=1}^{n} H(M = m^{[i]})$$

$$\lim_{p(m)\to p_{sat}} \frac{1}{n}H(M) < H(p_{sat})$$

$$\sup \frac{1}{n}H(M) = H(p_{sat}) \tag{7.11}$$

□

The result from Theorem 7.20 is useful because the calculation of the right hand side of the inequality (7.10) is trivial. In contrast, calculation of the left-hand side assuming the map belief is represented by a multi-variate Gaussian, requires maintaining the full map covariance matrix and computation of its determinant. This is not practical, since the map often has a dense belief representation and can be theoretically expanded unbounded (to a very large extent). In the following, we present some notable remarks and consequences of Theorem 7.20.

Remark 7.21. *The result from Theorem 7.20 also extends to continuous random variables and differential entropy.*

Remark 7.22. *Note that we do not assume any distribution for map points. The entropy can be calculated either with the assumption that the map points are normally distributed or treating them as Bernoulli random variables.*

Remark 7.23. *Since $0 < p_{sat} < 1$ and $H(p_{sat}) = H(1 - p_{sat})$, one saturation entropy can be set for the entire map.*

Corollary 7.24 (information gathering termination)**.** *Given a saturation entropy h_{sat}, the problem of search for information gathering for desired random variables $X_1, X_2, ..., X_n$ whose support is alphabet $\mathcal{X}$, can be terminated when $\frac{1}{n}\sum_{i=1}^{n} H(X_i) \leq h_{sat}$.*

Corollary 7.25 (Map exploration termination)**.** *The problem of autonomous robotic exploration for mapping can be terminated when $\frac{1}{n}\sum_{i=1}^{n} H(M = m^{[i]}) \leq H(p_{sat})$.*

The Corollary 7.24 generalizes the notion of exploration in the sense of information gathering. Therefore, regardless of the quantity of interest, we can provide a stopping criterion for the exploration mission. The Corollary 7.25 is of great importance for the classic robotic exploration for map completion problem as there is no need to resort to geometric frontiers with a specific cluster size to detect map completion. Another advantage of setting a threshold in the information space is the natural consideration of uncertainty in the estimation process before issuing a mission termination signal.

7.6 Results and Discussion

In this section, we examine the proposed algorithms in several scenarios. We first design experiments for comparison of information functions under various sensor parameters such as the number of beams and the sensor range, and their effects on the convergence of IIG. Although the primary objective of this experiment is to evaluate the performance of IIG using each information function, it can also be seen as a sensor selection problem. In other words, given the map of an environment and a number of sensors with different characteristics, we wish to select the sensor that has a reasonable balance of performance and cost. However, we do not emphasize this aspect as the main objective of the test as it requires more in-depth analysis of the problem and comparison with other available techniques.

The second experiment is what IIG is originally developed for. The robot explores an unknown environment; it needs to solve SLAM incrementally, estimate a dense occupancy map representation suitable for planning and navigation, and automatically detect the completion of an exploration mission. Therefore, the purpose of information gathering is map completion while maintaining accurate pose estimation, i.e. planning for estimation. We compare exploration experiments where the robot pose is estimated through a pose graph algorithm such as Pose SLAM,

and the map of the unknown environment is computed using the I-GPOM. Therefore, the state which includes the robot pose and the map is partially observable.

In the third scenario, we demonstrate another possible application of IIG in a lake monitoring experiment using experimental data collected by an Autonomous Surface Vehicle (ASV). This dataset is also used in the original RIG article (Hollinger and Sukhatme 2014) [10]. In the end, we also discuss the limitations of this work including our observations and conjectures. The algorithms were implemented in MATLAB and the code is publicly available [11].

7.6.1 Experimental Setup

We first briefly describe the experiment setup that is used in Subsections 7.6.2 and 7.6.3. The parameters for experiments and the information functions are listed in Table 7.1. The environments are constructed using a binary map of obstacles. For GPVR-based algorithms, the covariance function is Matérn ($\nu = 5/2$)

$$k_{VR} = \sigma_f^2 k_{\nu=5/2}(r) = \sigma_f^2 (1 + \frac{\sqrt{5}r}{l} + \frac{5r^2}{3l^2}) \exp(-\frac{\sqrt{5}r}{l}) \tag{7.12}$$

with hyperparameters that were learned prior to the experiments and are available in Table 7.1. The modified kernel in UGPVR algorithm was calculated using Gauss-Hermite quadrature with 11 sample points. The robot pose covariance was approximated using the worst-case uncertainty propagation and local linearization of the robot motion model using Equations (3.41) and (3.42). Each run was continued until the algorithm converges without any manual intervention. In case of multiple runs for an algorithm we report the mean and the standard error of outcomes. Furthermore, in all experiments, an unlimited budget is used to show the performance of each method in the limit and also examine our convergence criteria for IIG and exploration experiments (Corollary 7.25). The online parameters in Table 7.1 refer to exploration experiments and are used in Subsection 7.6.3.

[10] The dataset is publicly available on: `http://research.engr.oregonstate.edu/rdml/software`

[11] `https://github.com/MaaniGhaffari/sampling_based_planners`

TABLE 7.1: Parameters for IIG-tree experiments. "Online" parameters are only related to the exploration experiments.

Parameter	Symbol	Value
– General parameters:		
Occupied probability	p_{occ}	0.65
Unoccupied probability	p_{free}	0.35
Initial position	x_{init}	[10,2] m
Map resolution	δ_{map}	0.2 m
I_{RIC} threshold	δ_{RIC}	5e-4
I_{RIC} threshold (Online)	δ_{RIC}	1e-2
– MI-based parameters:		
Hit std	σ_{hit}	0.05 m
Short decay	λ_{short}	0.2 m
Hit weight	z_{hit}	0.7
Short weight	z_{short}	0.1
Max weight	z_{max}	0.1
Random weight	z_{rand}	0.1
Numerical integration resolution	s_z	2 m^{-1}
Saturation probability	p_{sat}	0.05
Saturation probability (Online)	p_{sat}	0.3
Occupied belief	b_{occ}	1.66
Unoccupied belief	b_{free}	0.6
– Covariance function hyperparameters:		
characteristic length-scale	l	3.2623 m
Signal variance	σ_f^2	0.1879
– Robot motion model:		
Motion noise covariance	$\boldsymbol{Q}$	$\mathrm{diag}(0.1\,\mathrm{m}, 0.1\,\mathrm{m}, 0.0026\,\mathrm{rad})^2$
Initial pose uncertainty	Σ_{init}	$\mathrm{diag}(0.4\,\mathrm{m}, 0.1\,\mathrm{m}, 0\,\mathrm{rad})^2$

7.6.2 Comparison of Information Functions

We now compare the proposed information functions by running IIG-tree using different sensor parameters. The map is known and initialized as an occupancy grid map by assigning each point the occupied or unoccupied probability according to its ground truth status. For GPVR-based methods, an initial variance map is set to the value of 1 for all points. In the first experiment, we varied the number of sensor beams by running each method for 30 times. The chosen number of beams are 10, 20, and 50. The compared information functions are MI, MIUB, GPVR, and UGPVR, and the results are presented in Table 7.2. The maximum sensor range is kept fixed at $r_{max} = 5\,\mathrm{m}$ and the convergence is detected when the average of penalized relative information contribution I_{RIC} over a window of size 30 drops below the threshold $\delta_{RIC} = 5e-4$. The total information gain/cost are calculated using the sum of all edges information/cost. Therefore, it denotes the total information/cost over the searched space and not a particular path. The reason

TABLE 7.2: Comparison of the information functions in offline IIG-tree experiments by increasing the sensor number of beams from 10 to 50. The figures are averaged over 30 experiments (mean ± standard error). For all experiments the following parameters are set in common $r_{max} = 5\,\text{m}$, $\delta_{RIC} = 5e-4$.

Number of beams n_z	10	20	50
Mutual information (MI)			
Planning time (min)	6.88 ± 0.16	8.44 ± 0.27	14.64 ± 0.81
Number of samples	911 ± 19	588 ± 13	435 ± 20
Number of nodes	402 ± 5	280 ± 5	181 ± 8
Total information gain (NATS)	1.4071e+04 ± 148.3	1.3624e+04 ± 173.8	1.2546e+04 ± 313.2
Total cost (m)	336.3 ± 3.8	258.5 ± 3.7	187.2 ± 7.7
Mutual information upper bound (MIUB)			
Planning time (min)	4.53 ± 0.16	6.95 ± 0.28	12.88 ± 0.51
Number of samples	1014 ± 24	629 ± 17	486 ± 15
Number of nodes	444 ± 8	303 ± 7	212 ± 5
Total information gain (NATS)	2.0472e+04 ± 169.2	1.9602e+04 ± 207.7	1.9881e+04 ± 403.1
Total cost (m)	358.3 ± 5.6	272.0 ± 5.0	211.5 ± 4.4
GP variance reduction (GPVR)			
Planning time (min)	9.31 ± 0.39	13.10 ± 0.41	15.07 ± 0.61
Number of samples	1677 ± 53	1135 ± 36	874 ± 36
Number of nodes	616 ± 15	477 ± 13	382 ± 11
Total information gain (NATS)	7840.6 ± 62.8	9123.4 ± 80.5	1.3111e+04 ± 176.9
Total cost (m)	490.5 ± 9.5	392.4 ± 7.9	320.6 ± 7.2
Uncertain GP variance reduction (UGPVR)			
Planning time (min)	19.75 ± 0.48	27.89 ± 0.94	47.79 ± 2.16
Number of samples	4746 ± 136	2633 ± 102	1828 ± 81
Number of nodes	1344 ± 28	913 ± 25	704 ± 22
Total information gain (NATS)	3197.6 ± 16.0	3731.3 ± 22.1	5557.5 ± 52.4
Total cost (m)	894.0 ± 16.3	621.4 ± 15.6	497.8 ± 12.5

is to avoid any possible randomness in path selection which can make the results biased. In other words, the results are independent of the path selection algorithm.

As it can be seen from Table 7.2, MIUB has the lowest runtime which is expected, followed by MI, GPVR, and UGPVR, respectively. However, MI has the fastest convergence speed with, approximately, half of the number of samples taken by GPVR. The calculation of MI can be more expensive than GPVR-based algorithms, but using sparse sensor observations (only 10 beams) and a very coarse numerical integration resolution it performs faster. For all the compared algorithms, as the number of beams increases, i.e. taking more observations, the computational time rises, and the number of samples/nodes reduces. An important observation is that incorporating the pose uncertainty in UGPVR leads to a slower convergence. This can be explained as a result of the reduction in the information content of each set of observations by adding uncertainty to

TABLE 7.3: Comparison of the information functions in offline IIG-tree experiments by increasing the sensor range from 5 m to 20 m (averaged over 30 experiments, mean ± standard error). For all experiments the following parameters are set in common $n_z = 10$; $\delta_{RIC} = 5e-4$.

Sensor range r_{max} (m)	5	10	20
Mutual information (MI)			
Planning time (min)	5.00 ± 0.13	6.60 ± 0.10	7.88 ± 0.11
Number of samples	979 ± 22	782 ± 21	780 ± 20
Number of nodes	423 ± 7	352 ± 8	360 ± 6
Total information gain (NATS)	1.4019e+04 ± 146.5	1.4525e+04 ± 177.9	1.5789e+04 ± 220.9
Total cost (m)	351.7 ± 4.8	307.8 ± 5.4	309.2 ± 4.0
Mutual information upper bound (MIUB)			
Planning time (min)	3.67 ± 0.10	3.56 ± 0.08	3.87 ± 0.11
Number of samples	1015 ± 23	813 ± 15	811 ± 20
Number of nodes	444 ± 7	368 ± 6	369 ± 7
Total information gain (NATS)	2.0414e+04 ± 161.7	1.9845e+04 ± 228.5	1.9730e+04 ± 272.9
Total cost (m)	360.1 ± 5.2	309.9 ± 4.2	311.6 ± 4.5
GP variance reduction (GPVR)			
Planning time (min)	6.20 ± 0.22	7.03 ± 0.24	6.60 ± 0.22
Number of samples	1745 ± 71	1393 ± 41	1248 ± 48
Number of nodes	636 ± 18	559 ± 13	523 ± 16
Total information gain (NATS)	7977.7 ± 56.4	7801.6 ± 71.0	7835.1 ± 62.5
Total cost (m)	501.7 ± 11.2	445.0 ± 7.5	422.7 ± 9.8
Uncertain GP variance reduction (UGPVR)			
Planning time (min)	20.52 ± 0.65	26.3 ± 0.68	26.68 ± 0.54
Number of samples	4994 ± 167	3661 ± 109	3385 ± 87
Number of nodes	1404 ± 35	1139 ± 25	1078 ± 19
Total information gain (NATS)	3174.8 ± 19.3	3254.2 ± 21.4	3271.0 ± 21.0
Total cost (m)	934.4 ± 22.3	765.2 ± 15.5	724.1 ± 12.3

the information gain calculation. Another interpretation is that the UGPVR algorithm is, relatively speaking, less greedy or more conservative for information gathering. MI in comparison to MIUB has more realistic information gain estimation which is reflected in less total cost on average. The latter results also confirm the advantage of maximizing the information gain rather than minimizing the entropy which is discussed in Krause et al. (2008).

In the second experiment, the number of beams is kept fixed at $n_z = 10$, but the sensor range is increased from 5 m to 20 m. The results shown in Table 7.3 are consistent with the previous test and the variations are even slighter. In fact, the first column of Table 7.3 is the repetition of the experiment with the similar parameters to the first column of Table 7.2, for another 30 runs [12]. Furthermore, the difference between $r_{max} = 10$ and $r_{max} = 20$ cases is marginal which shows

[12]The results are computed during several days using high-performance computing facilities at the University of Technology Sydney and the reported time is affected by other users activities. However, it can be used for relative comparison and other factors remain consistent.

increasing the sensor range more than 10 m does not improve the information gathering process in this environment. Overall, we can conclude that by changing the parameters the convergence of IIG using the relative information contribution notion remains consistent, and the speed of convergence varies reasonably with the parameters. We reiterate that the $\delta_{RIC} = 5e-4$ is used for all the results in Table 7.2 and 7.3 and this demonstrates a strong correlation with the planning time (horizon) for each information function, by ignoring the overhead computations due to the increase in n_z or r_{max}.

In Figure 7.1, an example of IIG-tree using MI information function in the Cave map (Howard and Roy 2003) is illustrated. The sensor range and the number of beams are set to 5 m and 10, respectively. Figure 7.1a shows the map of the environment, Figure 7.1b illustrates the IIG graph where the extracted paths are also shown and the most informative path is separated using a darker color. The convergence of I_{RIC} for both MI and MIUB is shown in Figure 7.1c and Figure 7.1d shows the calculated information gain in NATS where its diminishing return over time (as the number of samples grows) is evident.

The equivalent results for IIG-tree using GPVR and UGPVR information functions are shown in Figure 7.2 and Figure 7.3. It is interesting to observe that I_{RIC} converges to close values for MIUB and GPVR, but it has a lower value for MI as the estimated information gain is more realistic by taking the expectation over future measurements. For UGPVR information function, we can see that I_{RIC} is increased. From Figure 7.2d and Figure 7.3b, we can conclude that the overall trend is similar to the GPVR case. However, due to the incorporated pose uncertainty the amount of information gain is remarkably lower which explains the longer tail of information gain evolution before the convergence of the algorithm and the higher relative information contribution from each node. In other words, farther nodes have higher pose uncertainties. Therefore the discrimination between farther and nearer nodes from relative information contribution is less than the case where the pose uncertainty is ignored.

Finally, a practical conclusion can be that increasing the sensor range and the number of beams (observations) increases the computational time but does not necessarily make the corresponding information function superior to its coarser approximation. As long as the approximation can capture the essence of information gain estimation consistently, the search algorithm performs well. However, we should acknowledge that the conclusion is not complete, and the structural

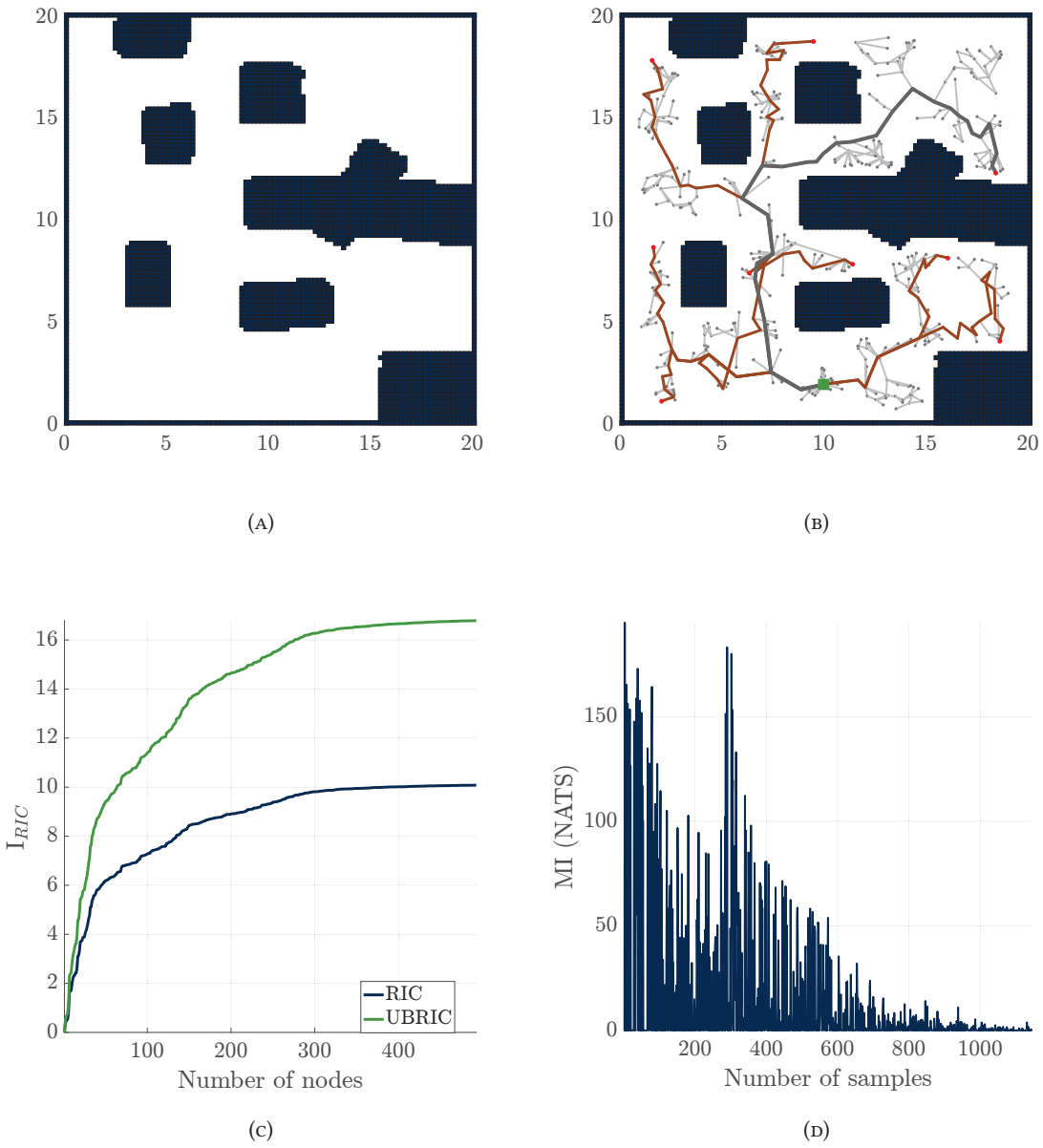


FIGURE 7.1: Results from running the IIG-tree algorithm using MI information function in the Cave map. (A) shows the map of the environment. (B) shows the IIG-tree graph and the selected paths, the most informative path is shown using a darker color. (C) is the convergence graph of the penalized relative information contribution I_{RIC} for MI and MIUB. (D) shows the evolution of the information gain calculation.

shape of the environment affects the result, e.g. the maximum and minimum perception field at different areas of the map. We use these results to select a sensor configuration for the next experiments. To speed up the exploration scenario, we select $n_z = 10$ and $r_{max} = 5\,\text{m}$ which lead to the fastest computational time with reasonable total information gain and cost values.

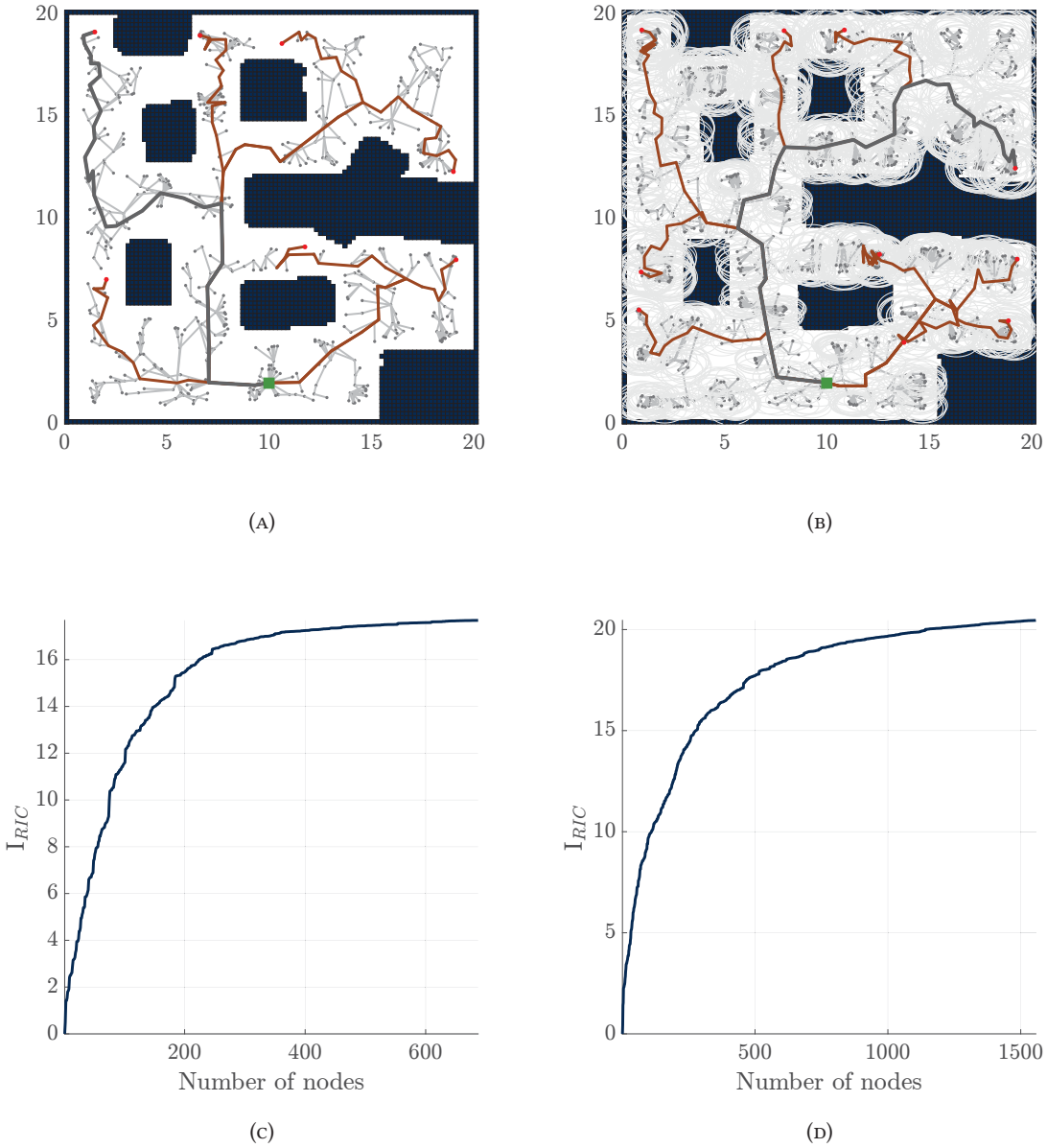
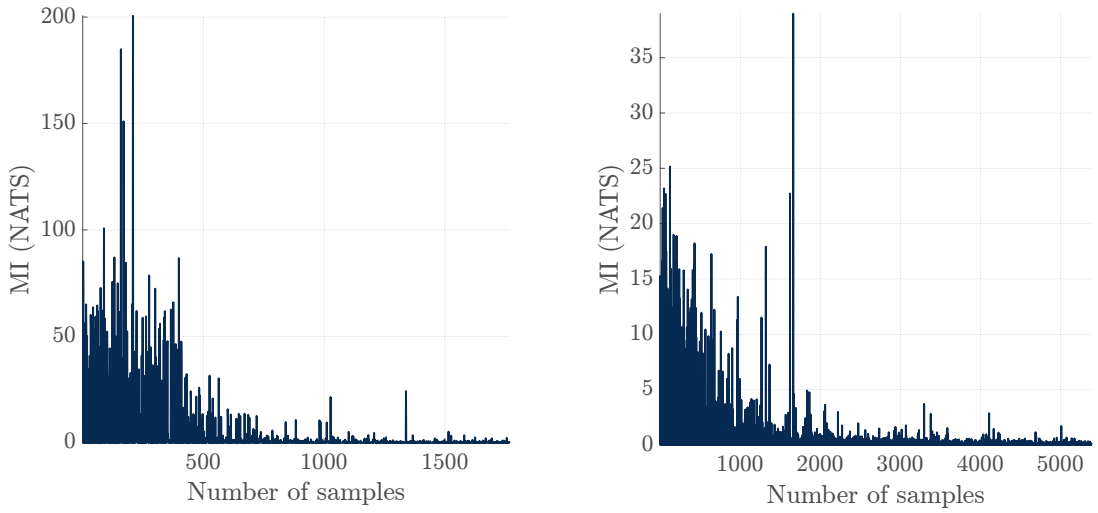


FIGURE 7.2: Results from running the IIG-tree algorithm using GPVR and UGPVR information functions in the Cave map. IIG-tree graph and the selected paths for (A) using GPVR information function, and (B) using UGPVR information function. The most informative path is shown using a darker color. The convergence graph of the penalized relative information contribution I_{RIC} for (C) GPVR, and (D) UGPVR.



(A) (B)

FIGURE 7.3: Evolution of the information gain calculation for (A) IIG-tree using GPVR information function, and (B) IIG-tree using UGPVR information function.

7.6.3 Robotic Exploration in Unknown Environments

In this section, we examine the proposed algorithms in the interesting and challenging scenario of autonomous robotic exploration in an unknown map. The robot does not have any prior information neither from the map nor its location. Therefore, it needs to solve the SLAM problem for localization and possibly the topological map of features (environment). Then building a dense map representation that shows occupied and unoccupied regions for planning and navigation. The robot pose and map are partially observable which makes the planning for a long horizon difficult. We solve the localization problem using Pose SLAM, the occupancy mapping using I-GPOM, and the planning using IIG-tree with different information functions for comparison. We also compare our results with Active Pose SLAM (APS) (Valencia et al. 2012) which is an information gain-based technique that considers explicit loop-closures by searching through nearby poses in the pose graph.

Figure 7.4 shows the statistical summary of the results from exploration experiments in the Cave environment. The results are from 30 exploration rounds in which each mission was terminated automatically using Corollary 7.25 and saturation probability $p_{sat} = 0.1$ [13]. The IIG-GPVR has the

[13]Note that MI-based information functions and map exploration termination condition have different saturation probabilities that do not necessarily have a similar value.

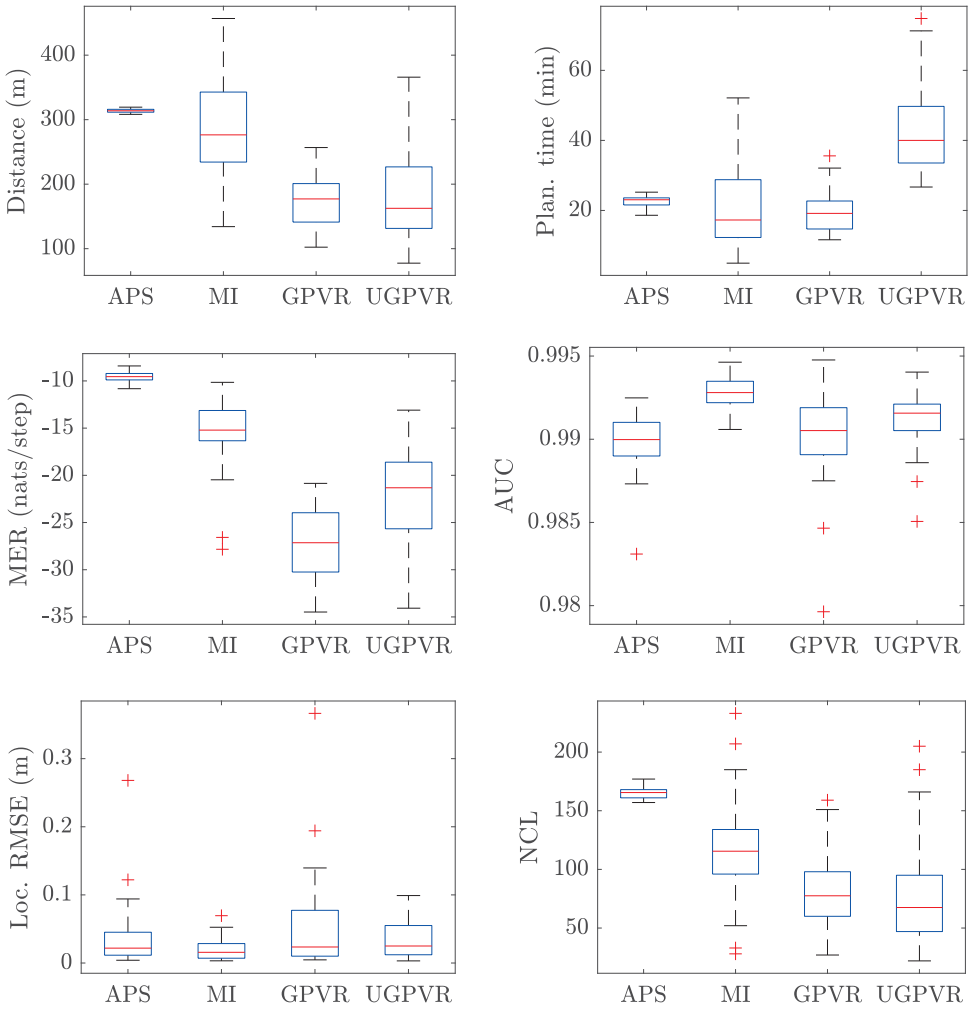


FIGURE 7.4: The box plots show the statistical summary of the comparison of different exploration strategies in the Cave dataset. The compared methods are APS, IIG-MI, IIG-GPVR, and IIG-UGPVR.

lowest travel distance, highest map entropy reduction rate, but, relatively speaking, the highest localization error. The improvement in the map entropy rate over our previous results on the same dataset using MI-based greedy exploration (Ghaffari Jadidi et al. 2015) is roughly more than 1.63 times which is surprisingly high for an algorithm that can preserve the localization accuracy better as well. IIG-UGPVR demonstrates a more conservative version of IIG-GPVR which by more planning time due to the pose uncertainty propagation provides better localization and map accuracies. This behavior is expected and shows the consistency between our problem definition and algorithmic development.

From another point of view, IIG-MI has the lowest localization error with a similar planning

TABLE 7.4: Lake monitoring experiments using IIG with GPVR and UGPVR information functions. For the comparison, the sensing range is varied from 5 m to 20 m. The results are averaged over 100 runs (mean ± standard error). For all experiments $\delta_{RIC} = 5e-4$.

Sensor range (m)	5	10	15	20
		IIG-GPVR		
Time (sec)	5.26 ± 0.16	2.92 ± 0.08	2.38 ± 0.09	2.32 ± 0.07
RMSE (dBm)	4.54 ± 0.05	4.28 ± 0.06	3.97 ± 0.06	3.61 ± 0.07
Number of samples	830.6 ± 18.7	507.9 ± 13.0	437.0 ± 15.6	428.1 ± 12.9
Number of nodes	826.1 ± 18.7	454.5 ± 12.8	359.9 ± 14.5	331.6 ± 12.7
Total information gain (NATS)	2.0817e+04 ± 160.1	2.4272e+04 ± 111.2	2.6207e+04 ± 133.1	2.7716e+04 ± 130.6
Total cost (Km)	6.77 ± 0.11	4.50 ± 0.09	3.67 ± 0.11	3.42 ± 0.10
		IIG-UGPVR		
Time (sec)	33.98 ± 1.30	25.93 ± 0.67	29.26 ± 0.98	53.69 ± 1.61
RMSE (dBm)	4.55 ± 0.05	4.26 ± 0.05	4.00 ± 0.06	3.56 ± 0.07
Number of samples	844.8 ± 19.8	496.0 ± 12.3	467.4 ± 15.4	413.0 ± 13.3
Number of nodes	839.2 ± 19.7	447.9 ± 12.4	389.1 ± 14.9	325.8 ± 12.8
Total information gain (NATS)	2.1268e+04 ± 176.7	2.4967e+04 ± 107.8	2.6925e+04 ± 120.6	2.8087e+04 ± 113.9
Total cost (Km)	6.81 ± 0.13	4.46 ± 0.09	3.91 ± 0.11	3.39 ± 0.10

time, but higher travel distance. The IIG-MI makes direct use of sensor model for information gain estimation. The fact that the sensor model for range-finders is an accurate model together with the correlation between the map and robot pose leads to implicit pose uncertainty reduction. This result reveals the fundamental difference between mutual information approximation using a direct method (taking the expectation over future measurements) and GPs which to best of our knowledge was not previously discussed in the literature.

7.6.4 Lake Monitoring Experiment

In this experiment, we demonstrate the performance of IIGs in a lake monitoring scenario. The ASV can localize using a GPS unit and a Doppler Velocity Log. The communication with the ground station is through an 802.11 wireless connection and at any location the wireless signal strength (WSS) can be measured in dBm. The Puddingstone Lake dataset is a publicly available dataset that includes about 2700 observations. The dataset is collected through a full survey of the lake area located at Puddingstone Lake in San Dimas, CA (Lat. 34.088854°, Lon. −117.810667°) (Hollinger and Sukhatme 2014). The objective is to find a trajectory for the robot to maintain a strong connectivity with the base station while taking physical samples in the lake.

The ground truth map of WSS is built using GP regression with a constant mean function, SE covariance function with automatic relevance determination (ARD) (Neal 1996), and a Gaussian

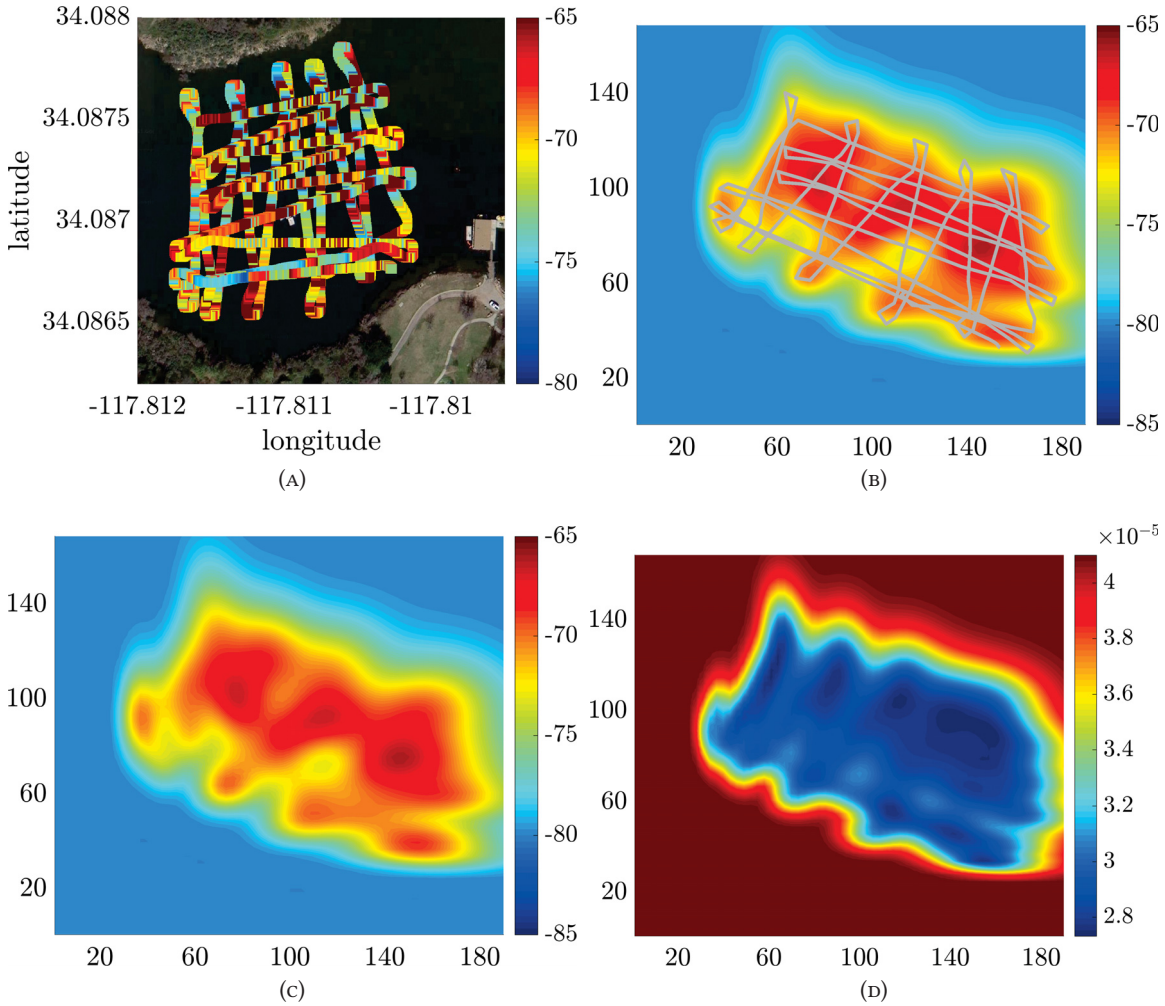


FIGURE 7.5: Lake monitoring scenario (a) satellite view of the lake and the survey trajectories using ASV, (b) the robot trajectories in metric scale on WSS map, (c) GP WSS mean surface (in dBm), and (d) GP WSS covariance surface.

likelihood function which makes the exact inference possible. Furthermore, it is well-known that, in line-of-sight scenarios, radio signals propagation can be characterized based on Friis free space model (Rappaport 1996; Goldsmith 2005). In this model, the signal attenuation is proportional to the logarithm of the distance. Therefore, to improve the regression accuracy we use logarithmic scales for input points during GP training and inference phases. The number of training points was down-sampled to 267 observations, and the surface was inferred using 3648 query points. We store the output using a kd-tree data structure to be able to perform fast online nearest neighbor inquiries within the robot sensing range. Figure 7.5 shows the satellite view of the lake area together with the survey trajectories and regressed maps that are used as a proxy for ground truth. Figure 7.5b shows the survey trajectories on the WSS surface where the longitudes and

latitudes are converted to their corresponding distances using the haversine formula. Figures 7.5c and 7.5d illustrate the GP WSS mean and covariance surfaces, respectively.

For information functions, we used Algorithms 13 and 14 as they are natural choices for scenarios involving spatial phenomena and environmental monitoring. The experiment is designed in such a way as that at any location the robot can take measurements within a sensing range from the grundtruth maps. This step replaces the raytracing operation in both algorithms. The modified kernel in Algorithm 14 was calculated using Gauss-Hermite quadrature with 11 sample points. The robot pose covariance was approximated using the worst-case uncertainty propagation and local linearization of the robot motion model [14]. Table 7.4 shows the comparison results between IIG-GPVR and IIG-UGPVR using several criteria including Root Mean-Squared Error (RMSE). To calculate the RMSE, we used the collected measurements along the most informative path extracted from the IIG graph using Algorithm 15 and rebuilt the GP WSS mean surface with the same resolution. We repeated the experiment 100 times with 5 m, 10 m, 15 m, and 20 m sensing ranges. The IIG-GPVR algorithm is faster and can compute the most informative trajectory in only a few seconds. This is promising for online applications where the robot needs to replan along the trajectory. Note that, as it is expected, by increasing the sensing range, the information gain rises and the total cost and RMSE decrease. However, this means it is assumed that in reality, the signal strength at any location has stronger correlations with its nearby locations. While this is true for the presented experiment, in a cluttered environment, it can be an unrealistic assumption.

Figure 7.6 shows the illustrative examples of IIG-GPVR with different sensing ranges. As the sensing range increases, the graph becomes gradually sparser. This effect can be understood from the fact that the information from a larger neighborhood is integrated into each node. Note that the objective is to explore the entire area while maintaining a strong wireless connectivity with the based station. Therefore, the robot does not need always to travel in the area with high signal strengths. However, it travels most of the time in the regions with strong connectivity. Figure 7.7 shows the same scenario using IIG-UGPVR. The main difference is that the robot behaves more conservatively and tends to return to the regions with strong connectivity. By traveling farther, the growth in pose uncertainty reduces the information gain of those areas.

[14] The pose uncertainties are not part of the original dataset. Hence, we calculate them by simulation.

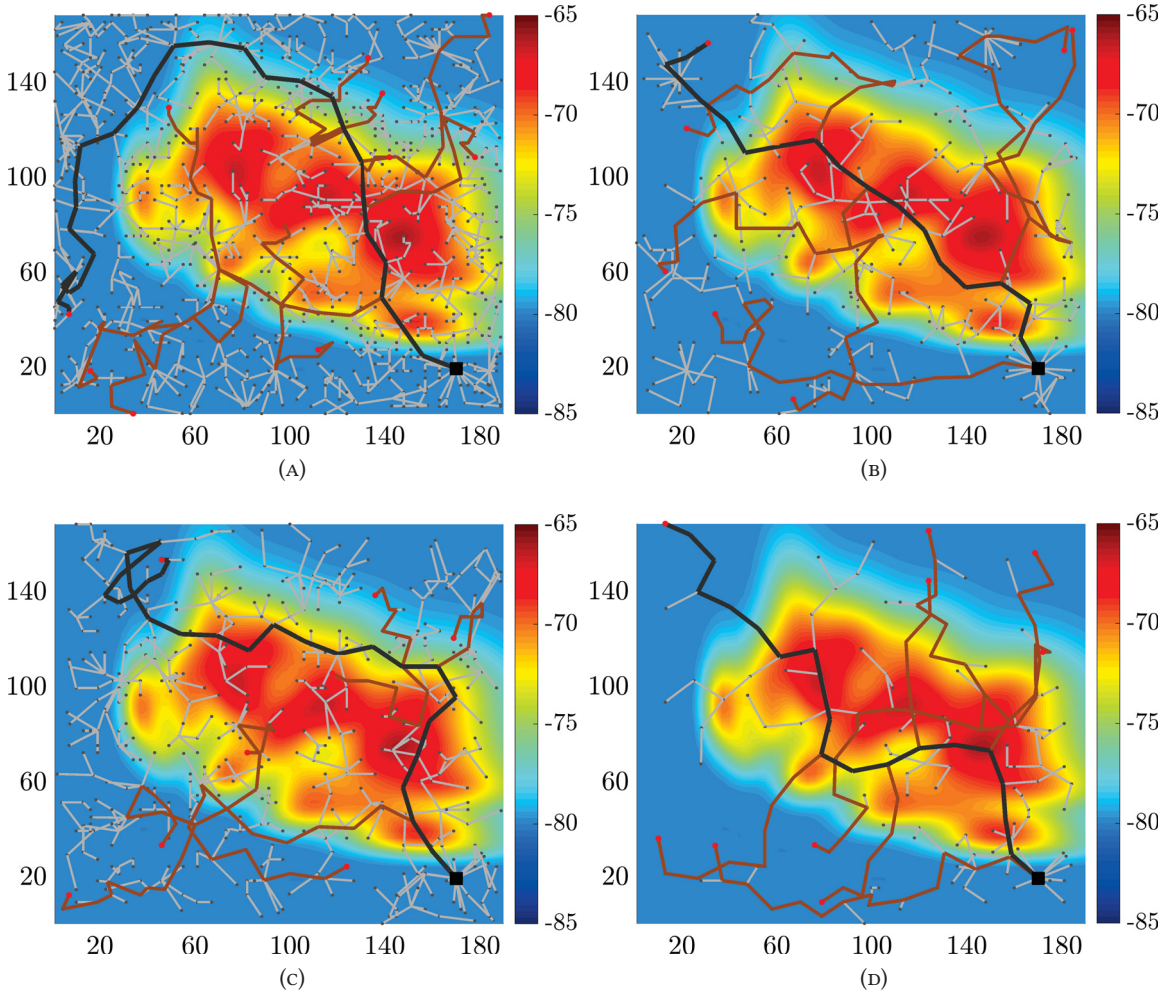


FIGURE 7.6: Informative motion planning for WSS monitoring in the lake area using IIG-GPVR with sensing range (a) 5 m (b) 10 m (c) 15 m, and (d) 20 m.

In both IIG-GPVR and IIG-UGPVR results, we can see that by increasing the sensing range the robot tends to explore farther distances which is a natural behavior, in Figures 7.6d and 7.7d.

7.6.5 Limitations and Observations

The proposed algorithms can provide an approximate solution to the robotic exploration problem in unknown environments as a basic navigation task as well as environmental monitoring tasks. Although IIG can be implemented for online tasks, it has the limitations of its ancestors. Therefore, violating the main assumptions, such as availability of free area near any point for graph expansion, can result in failure of the task. More conceptually, most of the present robotic

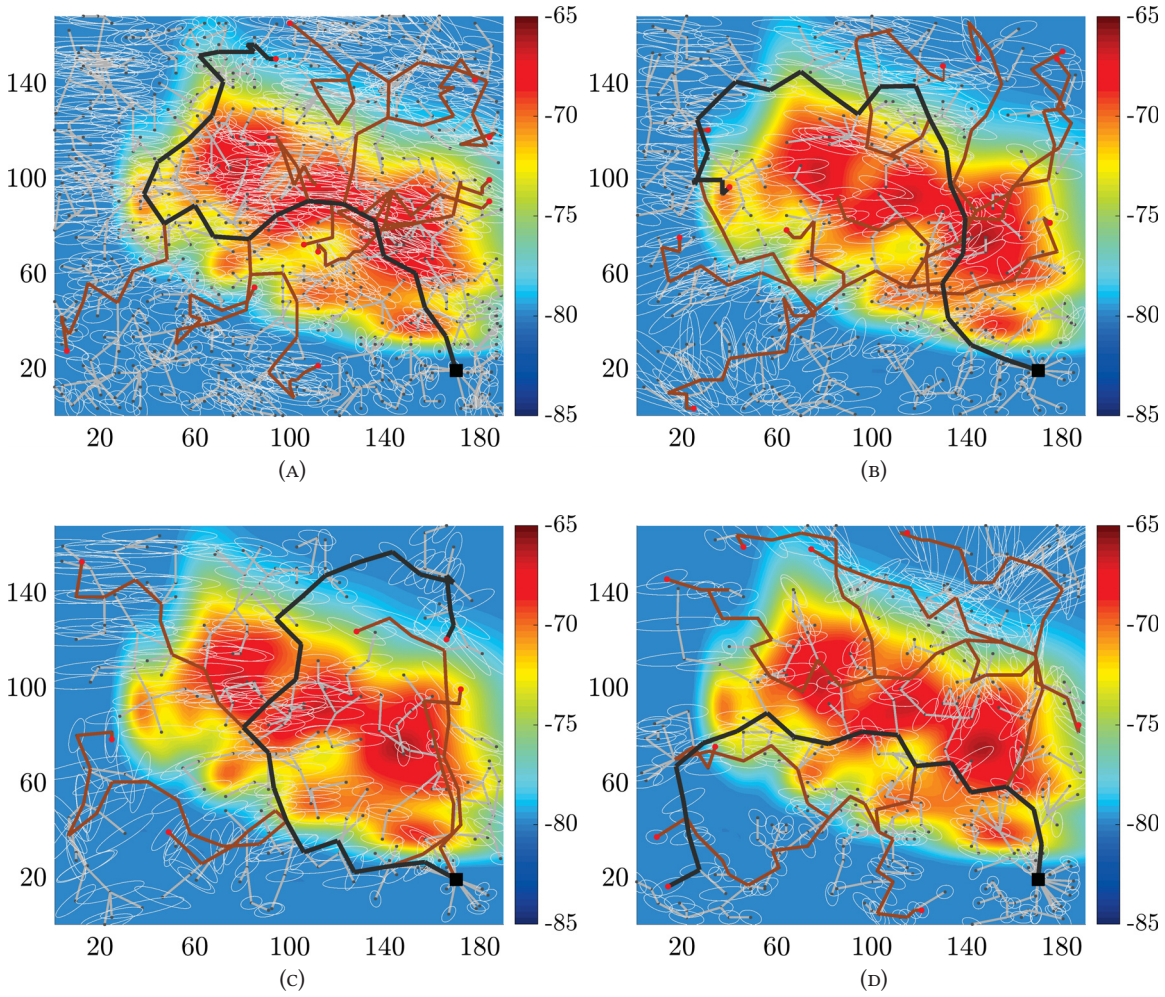


FIGURE 7.7: Informative motion planning for WSS monitoring in the lake area using IIG-UGPVR with sensing range (a) 5 m (b) 10 m (c) 15 m, and (d) 20 m. By incorporating pose uncertainties in planning, the informative trajectories are those in which not only the wireless connectivity is strong, but the pose uncertainty along the trajectories is minimized.

navigation algorithms are not truly adaptive as they commit to a decision once it is planned and replanning occurs only if the amount of computation is manageable, and often not with a high frequency. So there will be a window of time the robot acts based on the previously made decision. Although humans act in a similar fashion, we can make decisions more spontaneously (fully adaptive). This problem can be severe in environments that are highly dynamic as the robot is not as responsive as it is required.

To our experience, using a coarse approximation, i.e. less number of beams than the actual range-finder sensor results in not only faster search in the workspace and reducing the computational

time, but giving the chance to more samples to be candidates as part of the potential most informative path. This is a promising feature that the algorithm works reasonably well without near exact estimation of information quality. Moreover, uniform sampling produces reasonable results in the early stage, however as the graph grows, this sampling strategy becomes less efficient. Biasing the sampling towards directions that are less explored may lead to a faster convergence.

7.7 Chapter Summary

In this chapter, we developed a sampling-based planning algorithm for incremental robotic information gathering. The proposed algorithm is based on RIG and offers an information-theoretic convergence criterion using the notion of penalized relative information contribution. We formulated the problem as an information maximization problem subject to budget and state estimate constraints. The unique feature of this approach is that it can handle dense belief representations as well as the robot pose uncertainty. We proposed MI-based and GPVR-based information functions and their variants that can be used in both RIG and IIG frameworks. The proposed IIG-tree algorithm using a UGPVR information function considers pose uncertainties in planning and provides an approximate solution for finding maximally informative paths under partially observable state variables.

The proposed algorithms can directly be applied to environmental monitoring and robotic exploration tasks in unknown environments. The proposed algorithms also have potential applications in solving the sensor configuration selection problem that we did not fully investigate as it is beyond the scope of this work. Furthermore, the planning horizon does not need to be set as a number of steps. Instead, the information-theoretic representation of the planning horizon used here brings more flexibility for managing the convergence of the algorithm.

Chapter 8

Conclusion and Future Work

We studied the problem of robotic mapping and exploration in unknown environments in the presence of motion and sensing uncertainties. We started from Gaussian Processes occupancy mapping problem with known poses and extended the method to an incremental continuous occupancy mapping technique using the Bayesian committee machines. We showed that the accuracy of the proposed incremental mapping framework is close to that of the batch GPOM. The continuity of GPOM is exploited for a novel representation of geometric frontiers, and we showed that the GP-based mapping and exploration techniques are a competitor for traditional occupancy grid-based techniques. The primary motivations stemmed from the fact that high-dimensional map inference requires fewer observations to infer the map, leading to a faster map entropy reduction. Furthermore, we used the notion of macro-action to develop a greedy mutual information-based robotic exploration technique on the proposed novel map representation that takes into account all possible future measurements.

The problem of occupancy map building is extended to the case where the robot pose is uncertain and formulated using the expected kernel and expected sub-map approaches. While the expected kernel handles the input uncertainty within the GPs framework, the expected sub-map exploits the inherent property of stationary covariance functions for map inference in the local frame with deterministic inputs. Moreover, we showed that the WGPOM technique can deal with the nonlinear behavior of measurements through a nonlinear transformation which improves the

ability of GPs to learn complex structural shapes more accurately, especially, under uncertain inputs.

Since the original problem of robotic motion planning under uncertainty is NP-hard, based on the robotic information gathering concept, we developed a sampling-based incremental motion planning algorithm that has applications for robotic exploration and information gathering tasks. Through Gaussian Processes, the framework takes into account the state estimation uncertainty, and it can deal with dense belief representations. The developed algorithm does not discretize the robot action space, nor the state space. Instead, by sampling from the free workspace, the algorithm builds a graph representation of the most informative trajectories. Another interesting feature is the information-theoretic notion of the planning horizon which can be set as a convergence criterion rather than a number of steps. The planning horizon from the information gathering perspective has intuitive connections with the uncertainty minimization.

The developments in the early chapters are integrated into the proposed incremental information gathering framework in the form of mutual information-based information functions. The information functions algorithms exhibit different characteristics. For instance, MI-based functions are more suitable for exploration scenarios when a probabilistic sensor model is explicitly available and taking the expectation over future measurements is possible. On the other hand, GP-based functions are non-parametric and demonstrate promising properties for environmental monitoring tasks.

We would like to share some ideas that are natural extensions of this work. For the sake of clarity, we itemize them as follows.

- The algorithms in this thesis could be extended for multi-robot planning; exploiting the information gain available through wireless communication channels to manage the coordination between robots while a feedback controller satisfied kinodynamic constraints.
- We developed IIG-tree based on the RIG-tree algorithm. IIG could also be used to extend RIG-roadmap and RIG-graph. RIG-graph is asymptotically optimal and can asymptotically explore all possible budget-constraint trajectories, it produces a fully connected graph and is an interesting case to study. Therefore, IIG-graph would allow us to generate a unique

trajectory without choosing one from several, reducing the number steps required for incremental planning tasks such as robotic exploration.

– While we provide a procedural implementation of the proposed algorithms, and it suffices for research and comparison purposes, we believe the integration of the algorithms in open source libraries such as the Open Motion Planning Library (OMPL) (Sucan et al. 2012) has advantages for possible applications of this work.

– The GPVR-based algorithms could be used for active learning to model the quantity of interest online using GPs.

– IIG/RIG frameworks could be used for maximizing a multi-objective information function to perform multiple tasks concurrently. In a naïve approach, the objective function can be defined as the sum of information functions; however, if random variables from different information functions are correlated, e.g. through the robot pose, the total information gain calculation will not be accurate. Hence, developing a more systematic approach is an interesting avenue to follow.

– A sparse covariance matrix could be constructed using the Sparse covariance function. Exploiting the sparse structure of the covariance matrix for large-scale problems could be an interesting extension of this work for online implementations.

Appendix A

Mutual Information-based Exploration Results

The exploration results from extensive scenarios are presented in this appendix. The indoor experiments are focusing on comparison of the presented method with comparable exploratory methods available in the literature, whereas the outdoor experiment demonstrates the scalability. An average improvement over the standard and state-of-the-art exploratory methods by more than 20% and 13% in travel distance and map entropy reduction rate, respectively, while maintaining the localization Mean Squared Error (MSE) 36% lower, can be clearly seen in Tables A.1 and A.2. The localization MSE was computed at the end of each experiment by the difference between the robot traveled path, i.e. estimated poses and ground truth poses. The required parameters for the beam-based mixture measurement model, frontier maps, and MI maps computations are listed in Table 5.2.

The experiments include exhaustive comparison among the original Nearest Frontier (NF) (Yamauchi 1997), APS (Valencia et al. 2012), GPNF, and GPMI exploration approaches. In order to maintain NF and APS methods in their original frameworks, the results are computed using OGMs while in COM-based methods the developed COM and the probabilistic frontier map are employed. In all of the presented results, Pose SLAM (Ila et al. 2010) is included as the backbone to provide localization data together with the number of closed loops. Additionally, regardless of the exploration method, the same set of Pose SLAM parameters was used in each

environment. The simulated robot and its sensors, i.e. the odometric and laser range-finder, provide required sensory inputs for Pose SLAM and the environment is built by loading a binary image. The odometric and laser range-finder sensors noise covariances were set to $\Sigma_u = \text{diag}(0.1\,\text{m}, 0.1\,\text{m}, 0.0026\,\text{rad})^2$ and $\Sigma_y = \text{diag}(0.05\,\text{m}, 0.05\,\text{m}, 0.0017\,\text{rad})^2$, respectively. The robot started the experiments with an initial pose uncertainty of $\Sigma_0 = \text{diag}(0.1\,\text{m}, 0.1\,\text{m}, 0.09\,\text{rad})^2$ and laser beams were simulated through ray-casting operation over the ground truth map using the true robot pose.

The proposed approach is demonstrated with exploratory simulations in three mapping environments. The indoor datasets include the Cave and Freiburg maps (Howard and Roy 2003) and the outdoor dataset is a parking area as an experiment on a larger scale (16 times on average in the area) to demonstrate the scalability of the presented approach. The Cave map represents a simple hand-drawn environment with a few rough obstacles, whereas the Freiburg map is computed from a real LMS-laser log taken with a Pioneer2 robot at the University of Freiburg (building 079 AIS-Lab), and contains many rooms and disconnected obstacles which make for a challenging environment to explore. The implementation has been developed in MATLAB and GP computations have been implemented using the open source GP library in Rasmussen and Williams (2006).

A.1 Indoor Experiments

Table A.1 summarizes the exploration results in the indoor environments. The figures are averaged over 30 repetitions of the experiments in order to demonstrate repeatability and reliability of each method. To achieve the best outcome for all of the compared methods, in the nearest frontier methods (NF and GPNF), frontiers closer than 2 m to the current robot pose are ignored. Furthermore, only frontiers with the size larger than 10 cells are considered valid in NF and APS. For GPNF and GPMI, a threshold was set for the frontier map as the termination condition of an experiment. These conditions lead to avoiding excessive search and fluctuation in a small area and exploring the whole map faster.

Despite the common belief in the literature, classic NF performs reasonably satisfactorily in terms of minimum requirements for an exploration method. However, NF does not contemplate any

Table A.1: Comparison of the exploration strategies in the indoor datasets (averaged over 30 experiments, mean ± standard error)

Cave environment 20 m ×20 m; map resolution: 0.2 m; frontier map threshold: 0.59				
	NF	APS	GPNF	GPMI
Travel distance (m)	109.12 ± 4.72	317.83 ± 0.40	108.60 ± 7.64	**100.54 ± 3.46**
Exploration time (min)	14.50 ± 0.48	18.58 ± 0.28	**5.40 ± 0.43**	11.25 ± 0.38
Map ent. rate (NATS/step)	-14.1576 ± 0.4683	-10.2131 ± 0.1500	-15.1586 ± 0.4250	**-16.5754 ± 0.4905**
Localization MSE (m^2)	0.3392 ± 0.0629	0.1234 ± 0.0348	0.0930 ± 0.0189	**0.0784 ± 0.0135**
Number of closed loops	17.10 ± 2.17	**157.30 ± 1.28**	31.30 ± 5.08	25.60 ± 2.11
Freiburg environment 40 m ×15 m; map resolution: 0.2 m; frontier map threshold: 0.57				
	NF	APS	GPNF	GPMI
Travel distance (m)	265.26 ± 10.59	n/a	170.37 ± 6.20	**154.46 ± 5.39**
Exploration time (min)	57.29 ± 1.25	n/a	18.23 ± 0.87	**15.13 ± 0.90**
Map ent. rate (NATS/step)	-8.2347 ± 0.1582	n/a	-12.6333 ± 0.3509	**-13.0932 ± 0.3100**
Localization MSE (m^2)	0.6135 ± 0.1432	n/a	0.3257 ± 0.0470	**0.2035 ± 0.0256**
Number of closed loops	**17.00 ± 1.11**	n/a	9.07 ± 0.64	7.87 ± 0.60

form of uncertainty reduction which explains the highest localization error in Table A.1. APS tries to minimize the approximated joint entropy of the map and robot pose which results in completion of the experiment with a lower localization error. However, by aiming for the most informative goal together with the replanning policy, according to the utility function, the robot may fluctuate among several areas of the map. The increased travel distance and significantly higher number of closed loops imply over-visiting of some already known areas. The results show that grid-based mapping does not give sufficient support for information gain-based exploration as the robot may travel a long distance to approach an informative area and meanwhile collect sparse measurement along its way. This is where the importance of GP mapping becomes highlighted through handling sparse measurements and fast map entropy reduction. In other words, in many cases, the robot does not need to revisit partially observed areas; and the correlation between points in the map helps in reducing the map entropy at a faster rate.

GPNF and GPMI exploit COM for mapping, exploration, and planning providing a common ground for both travel distance and uncertainty reductions. GP-based methods handle sparse sensor measurements and by learning the environment's structural dependencies define distributions over the map points occupancy and also frontier points. The significant increase in the map entropy reduction rate while maintaining the localization error low is due to these facts and can be concluded from the data in Table A.1. GPMI through the developed utility function tries to maximize the MI between map and future measurements. It demonstrates a better long term decision making, since it has the fastest map entropy reduction rate as the MI map suggests.

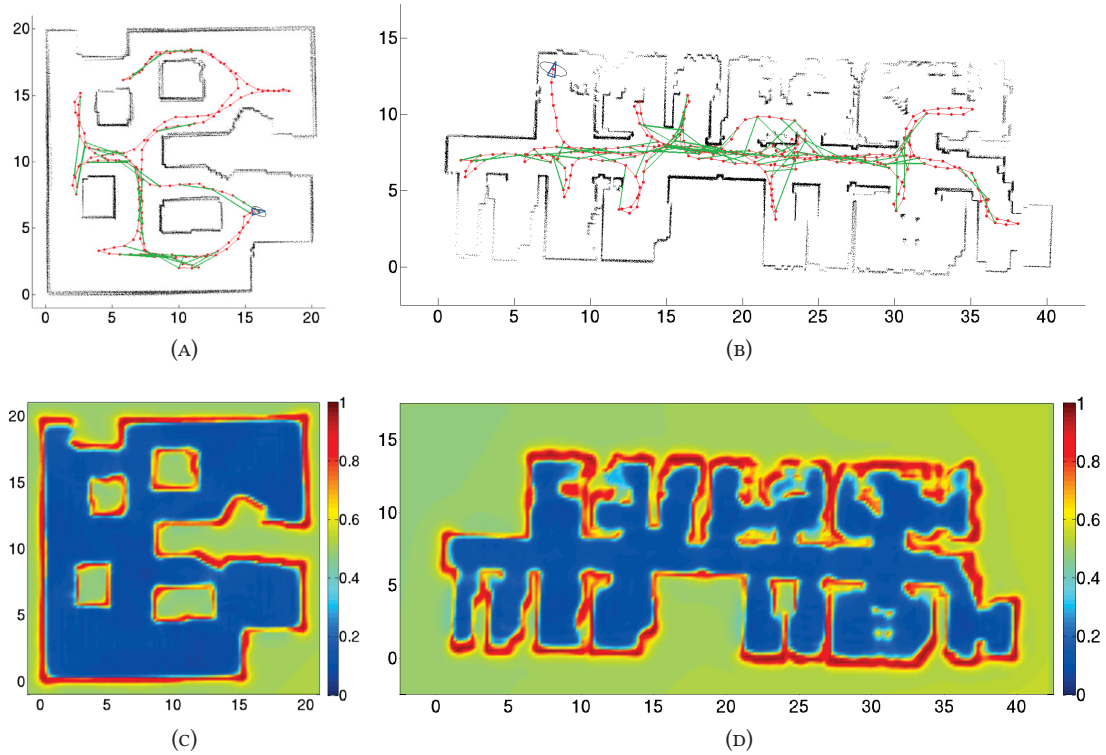


FIGURE A.1: MI-based exploration in Cave and Freiburg environments. (a) and (b) Pose SLAM maps, (c) and (d) continuous occupancy maps. In (a) and (b), red curves are the robot path and green lines indicate loop-closures. The continuous occupancy maps show the occupancy probability at each location where mid-probability value of 0.5 represents unknown points. Map dimensions are in meters.

The developed MI surface is a suitable cost map for planners such as A^* as demonstrated in the results. Although the map is one-step look-ahead it is assumed to be fixed during the planning time. Furthermore, traveling through uncertain traversable regions facilitates closing informative loops in the SLAM process by taking overlapping observations from diverse orientations. Figure A.1 illustrates the indoor exploration results with the GPMI method in Cave and Freiburg datasets.

A.2 Outdoor Experiments

The outdoor scenario consists of a large parking area, approximately 16 times larger than the earlier indoor setting. Figure A.2 shows the exploration results with the GPMI method. The environment is challenging, not only due to its size, but also for the long sections with few

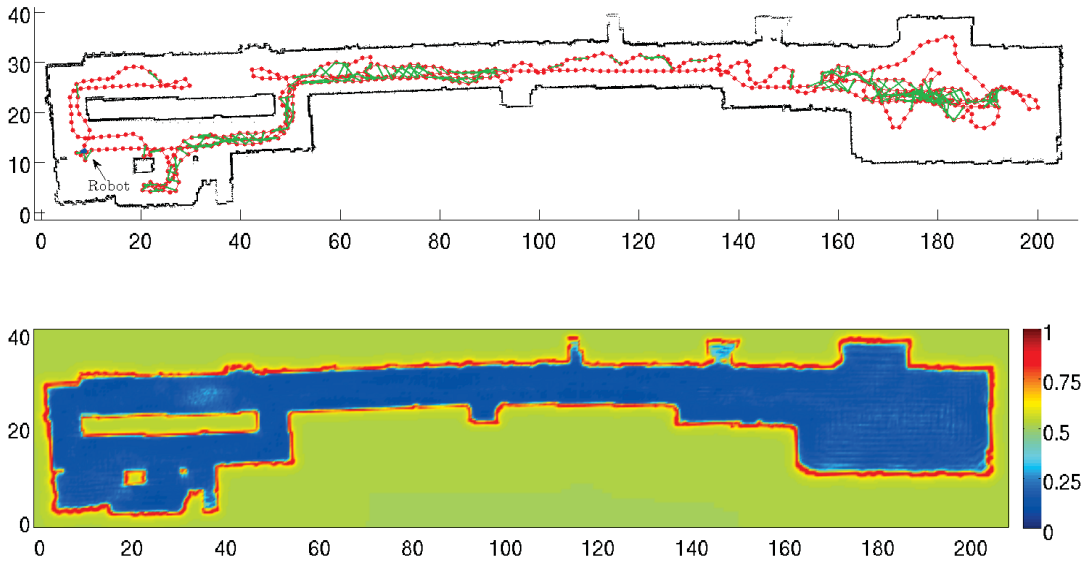


FIGURE A.2: MI-based exploration in an outdoor parking area. This map, on average, is 16 times larger than the indoor maps in area and demonstrates the scalability of the presented method. The lack of distinctive features in many region makes closing informative loops extremely challenging which can easily lead to the localization inconsistency in such a large environment. Map dimensions are in meters.

TABLE A.2: Comparison of the GPNF and GPMI in the outdoor dataset (averaged over 10 experiments, mean ± standard error)

Outdoor parking environment 210 m ×42 m; map resolution: 0.4 m frontier map threshold: 0.53		
	GPNF	GPMI
Travel distance (m)	634.44 ± 36.54	**594.94 ± 53.53**
Exploration time (min)	**20.51 ± 1.00**	38.65 ± 2.17
Map ent. rate (NATS/step)	-72.8809 ± 4.0572	**-74.1553 ± 3.8043**
Localization MSE (m^2)	2.5288 ± 0.4777	**1.7357 ± 0.4664**
Number of closed loops	**34.50 ± 4.83**	24.10 ± 8.89

distinguishing features. Closing informative loops is non-trivial and moving towards frontiers as they emerge along the long straight section in the middle of the map significantly increases robot pose uncertainty. Table A.2 presents the comparative results of running GPNF and GPMI in this dataset. It can be seen how the latter performs slightly better in regards to travel distance, map entropy rate, and localization error, while computational cost is slightly worse. A longer run experiment such as this reveals how the lack of an explicit utility term for loop-closure actions results in statistically closer figures.

Bibliography

Agha-mohammadi Aa, Chakravorty S and Amato NM (2014) FIRM: Sampling-based feedback motion-planning under motion uncertainty and imperfect measurements. *The Int. J. Robot. Res.* 33(2): 268–304.

Amigoni F and Caglioti V (2010) An information-based exploration strategy for environment mapping with mobile robots. *Robot. Auton. Syst.* 58(5): 684–699.

Arya S and Mount DM (1993) Approximate nearest neighbor queries in fixed dimensions. In: *Proc. 4th Ann. ACM-SIAM Symposium on Discrete Algorithms (SODA)*. pp. 271–280.

Astrom K (1965) Optimal control of markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications* 10(1): 174–205.

Bagon S (2009) Matlab class for ANN. URL `http://www.wisdom.weizmann.ac.il/~bagon/matlab.html`.

Bailey T, Nieto J, Guivant J, Stevens M and Nebot E (2006a) Consistency of the EKF-SLAM algorithm. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, pp. 3562–3568.

Bailey T, Nieto J and Nebot E (2006b) Consistency of the FastSLAM algorithm. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 424–429.

Bajcsy R (1988) Active perception. *Proceedings of the IEEE* 76(8): 966–1005.

Bertsekas DP (1995) *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, MA.

Binney J, Krause A and Sukhatme GS (2013) Optimizing waypoints for monitoring spatiotemporal phenomena. *The Int. J. Robot. Res.* 32(8): 873–888.

Binney J and Sukhatme GS (2012) Branch and bound for informative path planning. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 2147–2154.

Borrmann D, Elseberg J, Lingemann K, Nüchter A and Hertzberg J (2008) Globally consistent 3d mapping with scan matching. *Robot. Auton. Syst.* 56(2): 130–142.

Bosse M, Newman P, Leonard J and Teller S (2004) Simultaneous localization and map building in large-scale cyclic environments using the atlas framework. *The Int. J. Robot. Res.* 23(12): 1113–1139.

Bosse M and Zlot R (2009) Continuous 3d scan-matching with a spinning 2d laser. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 4312–4319.

Bourgault F, Makarenko AA, Williams SB, Grocholsky B and Durrant-Whyte HF (2002) Information based adaptive robotic exploration. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, volume 1. IEEE, pp. 540–545.

Bry A and Roy N (2011) Rapidly-exploring random belief trees for motion planning under uncertainty. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 723–730.

Burgard W, Moors M, Fox D, Simmons R and Thrun S (2000) Collaborative multi-robot exploration. In: *Proc. IEEE Int. Conf. Robot Automat.*, volume 1. pp. 476–481.

Cadena C, Carlone L, Carrillo H, Latif Y, Scaramuzza D, Neira J, Reid I and Leonard JJ (2016) Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* 32(6): 1309–1332.

Calandra R, Peters J, Rasmussen CE and Deisenroth MP (2014) Manifold Gaussian processes for regression. *arXiv preprint arXiv:1402.5876* .

Carlone L, Du J, Ng MK, Bona B and Indri M (2010) An application of kullback-leibler divergence to active SLAM and exploration with particle filters. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, pp. 287–293.

Charrow B, Kumar V and Michael N (2014) Approximate representations for multi-robot control policies that maximize mutual information. *Auton. Robot* 37(4): 383–400.

Charrow B, Liu S, Kumar V and Michael N (2015) Information-theoretic mapping using Cauchy-Schwarz quadratic mutual information. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 4791–4798.

Cover TM and Thomas JA (1991) *Elements of information theory*. John Wiley & Sons.

Cummins M and Newman P (2008) Fab-map: Probabilistic localization and mapping in the space of appearance. *The Int. J. Robot. Res.* 27(6): 647–665.

Davis PJ and Rabinowitz P (1984) *Methods of numerical integration*. Academic Press.

Dellaert F and Kaess M (2006) Square root sam: Simultaneous localization and mapping via square root information smoothing. *The Int. J. Robot. Res.* 25(12): 1181–1203.

Dhariwal A, Sukhatme GS and Requicha AA (2004) Bacterium-inspired robots for environmental monitoring. In: *Proc. IEEE Int. Conf. Robot Automat.*, volume 2. IEEE, pp. 1436–1443.

Dijkstra EW (1959) A note on two problems in connexion with graphs. *Numerische mathematik* 1(1): 269–271.

Diosi A and Kleeman L (2005) Laser scan matching in polar coordinates with application to SLAM. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, pp. 3317–3322.

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

Doucet A, De Freitas N, Murphy K and Russell S (2000) Rao-blackwellised particle filtering for dynamic bayesian networks. In: *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., pp. 176–183.

Dunbabin M and Marques L (2012) Robots for environmental monitoring: Significant advancements and applications. *IEEE Robotics & Automation Magazine* 19(1): 24–39.

Durrant-Whyte H and Bailey T (2006) Simultaneous localization and mapping: part I. *Robotics & Automation Magazine, IEEE* 13(2): 99–110.

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

Estrada C, Neira J and Tardós JD (2005) Hierarchical SLAM: Real-time accurate mapping of large environments. *IEEE Trans. Robot.* 21(4): 588–596.

Eustice RM, Singh H and Leonard JJ (2006) Exactly sparse delayed-state filters for view-based SLAM. *IEEE Trans. Robot.* 22(6): 1100–1114.

Fawcett T (2006) An introduction to ROC analysis. *Pattern recognition letters* 27(8): 861–874.

Feder HJS, Leonard JJ and Smith CM (1999) Adaptive mobile robot navigation and mapping. *The Int. J. Robot. Res.* 18(7): 650–668.

Gan S, Yang K and Sukkarieh S (2009) 3D online path planning in a continuous Gaussian process occupancy map. In: *Proc. Australian Conf. Field Robotics.*

Ghaffari Jadidi M, Miro JV, Valencia R and Andrade-Cetto J (2014) Exploration on continuous Gaussian process frontier maps. In: *Proc. IEEE Int. Conf. Robot Automat.* pp. 6077–6082.

Ghaffari Jadidi M, Valls Miro J and Dissanayake G (2015) Mutual information-based exploration on continuous occupancy maps. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* pp. 6086–6092.

Ghaffari Jadidi M, Valls Miro J and Dissanayake G (2017) Warped Gaussian processes occupancy mapping with uncertain inputs. *IEEE Robotics and Automation Letters* PP(99): 1–1. doi: 10.1109/LRA.2017.2651154.

Ghaffari Jadidi M, Valls Miro J, Valencia Carreño R, Andrade-Cetto J and Dissanayake G (2013a) Exploration in information distribution maps. In: *RSS Workshop on Robotic Exploration, Monitoring, and Information Content.*

Ghaffari Jadidi M, Valls Miro J, Valencia Carreño R, Andrade-Cetto J and Dissanayake G (2013b) Exploration using an information-based reaction-diffusion process. In: *Australasian Conf. Robot. Automat.*

Girard A (2004) *Approximate methods for propagation of uncertainty with Gaussian process models.* PhD Thesis, University of Glasgow.

Girard A, Rasmussen CE, nonero Candela JQ and Murray-Smith R (2003) Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting. In: *Advances in Neural Information Processing Systems.* pp. 545–552.

Goldberg PW, Williams CK and Bishop CM (1998) Regression with input-dependent noise: A Gaussian process treatment. In: *Advances in Neural Information Processing Systems*. pp. 493–499.

Goldsmith A (2005) *Wireless communications*. Cambridge university press.

González-Banos H and Latombe J (2002) Navigation strategies for exploring indoor environments. *The Int. J. Robot. Res.* 21(10-11): 829–848.

Grisetti G, Rizzini DL, Stachniss C, Olson E and Burgard W (2008) Online constraint network optimization for efficient maximum likelihood map learning. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 1880–1885.

Guestrin C, Krause A and Singh AP (2005) Near-optimal sensor placements in Gaussian processes. In: *Proc. Int. Conf. Machine learning*. ACM, pp. 265–272.

Guivant JE and Nebot EM (2003) Solving computational and memory requirements of feature-based simultaneous localization and mapping algorithms. *Robotics and Automation, IEEE Transactions on* 19(4): 749–755.

Hadsell R, Bagnell JA, Huber D and Hebert M (2010) Space-carving kernels for accurate rough terrain estimation. *The Int. J. Robot. Res.* 29(8): 981–996.

He R, Brunskill E and Roy N (2010) Puma: Planning under uncertainty with macro-actions. In: *AAAI*.

Hensman J, Fusi N and Lawrence ND (2013) Gaussian processes for big data. *arXiv preprint arXiv:1309.6835* .

Ho KL and Newman P (2007) Detecting loop closure with scene sequences. *Int. J. Computer Vision* 74(3): 261–286.

Hollinger GA (2015) Long-horizon robotic search and classification using sampling-based motion planning. In: *Robotics: Science and Systems*.

Hollinger GA and Sukhatme GS (2013) Sampling-based motion planning for robotic information gathering. In: *Robotics: Science and Systems*.

Hollinger GA and Sukhatme GS (2014) Sampling-based robotic information gathering algorithms. *The Int. J. Robot. Res.* 33(9): 1271–1287.

Hornung A, Wurm KM, Bennewitz M, Stachniss C and Burgard W (2013) OctoMap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots* 34(3): 189–206.

Horsch T, Schwarz F and Tolle H (1994) Motion planning with many degrees of freedom-random reflections at c-space obstacles. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 3318–3323.

Howard A and Roy N (2003) The robotics data set repository (Radish). URL `http://radish.sourceforge.net`.

Huang S and Dissanayake G (2007) Convergence and consistency analysis for extended kalman filter based SLAM. *IEEE Trans. Robot.* 23(5): 1036–1049.

Huynh VA and Roy N (2009) icLQG: combining local and global optimization for control in information space. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 2851–2858.

Ila V, Andrade-Cetto J, Valencia R and Sanfeliu A (2007) Vision-based loop closing for delayed state robot mapping. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, pp. 3892–3897.

Ila V, Porta J and Andrade-Cetto J (2010) Information-based compact Pose SLAM. *IEEE Trans. Robot.* 26(1): 78–93.

Indelman V, Carlone L and Dellaert F (2015) Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments. *The Int. J. Robot. Res.* 34(7): 849–882.

Jordan MI, Ghahramani Z, Jaakkola TS and Saul LK (1999) An introduction to variational methods for graphical models. *Machine learning* 37(2): 183–233.

Juliá M, Gil A and Reinoso O (2012) A comparison of path planning strategies for autonomous exploration and mapping of unknown environments. *Auton. Robot* 33(4): 427–444.

Juliá M, Reinoso Ó, Gil A, Ballesta M and Payá L (2010) A hybrid solution to the multi-robot integrated exploration problem. *Eng. Appl. Artif. Intell.* 23(4): 473–486.

Julian BJ, Karaman S and Rus D (2014) On mutual information-based control of range sensing robots for mapping applications. *The Int. J. Robot. Res.* 33(10): 1375–1392.

Julier SJ and Uhlmann JK (1997) New extension of the kalman filter to nonlinear systems. In: *AeroSense'97*. Int. Society for Optics and Photonics, pp. 182–193.

Kaelbling LP, Littman ML and Cassandra AR (1998) Planning and acting in partially observable stochastic domains. *Artificial intelligence* 101(1): 99–134.

Kaess M, Johannsson H, Roberts R, Ila V, Leonard J and Dellaert F (2011) isam2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 3281–3288.

Kaess M, Ranganathan A and Dellaert F (2008) isam: Incremental smoothing and mapping. *IEEE Trans. Robot.* 24(6): 1365–1378.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *The Int. J. Robot. Res.* 30(7): 846–894.

Kavraki LE, Švestka P, Latombe JC and Overmars MH (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *Robotics and Automation, IEEE Transactions on* 12(4): 566–580.

Keidar M and Kaminka GA (2014) Efficient frontier detection for robot exploration. *The Int. J. Robot. Res.* 33(2): 215–236.

Kersting K, Plagemann C, Pfaff P and Burgard W (2007) Most likely heteroscedastic Gaussian process regression. In: *Proc. Int. Conf. Machine learning*. ACM, pp. 393–400.

Kim A and Eustice RM (2015) Active visual SLAM for robotic area coverage: Theory and experiment. *The Int. J. Robot. Res.* 34(4-5): 457–475.

Kim S and Kim J (2012) Building occupancy maps with a mixture of Gaussian processes. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 4756–4761.

Kim S and Kim J (2013a) Continuous occupancy maps using overlapping local Gaussian processes. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, pp. 4709–4714.

Kim S and Kim J (2013b) Occupancy mapping and surface reconstruction using local Gaussian processes with Kinect sensors. *IEEE Transactions on Cybernetics* 43(5): 1335–1346.

Kim S and Kim J (2015) GPmap: A unified framework for robotic mapping based on sparse Gaussian processes. In: *Field and Service Robotics*. Springer, pp. 319–332.

Ko J and Fox D (2009) GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. *Auton. Robot* 27(1): 75–90.

Konolige K, Grisetti G, Kümmerle R, Burgard W, Limketkai B and Vincent R (2010) Efficient sparse pose adjustment for 2d mapping. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, pp. 22–29.

Krause A and Guestrin C (2005) Near-optimal value of information in graphical models. In: *Conference on Uncertainty in Artificial Intelligence (UAI)*.

Krause A, Singh A and Guestrin C (2008) Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *The Journal of Machine Learning Research* 9: 235–284.

Kurniawati H, Du Y, Hsu D and Lee WS (2011) Motion planning under uncertainty for robotic tasks with long time horizons. *The Int. J. Robot. Res.* 30(3): 308–323.

Kurniawati H, Hsu D and Lee WS (2008) Sarsop: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: *Robotics: Science and Systems*.

Lan X and Schwager M (2013) Planning periodic persistent monitoring trajectories for sensing robots in Gaussian random fields. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 2415–2420.

Lang T, Plagemann C and Burgard W (2007) Adaptive non-stationary kernel regression for terrain modeling. In: *Robotics: Science and Systems*.

Latombe JC (1991) *Robot motion planning*, volume 124. Springer US.

Lau H (2003) Behavioural approach for multi-robot exploration. In: *Australasian Conf. Robot. Automat.*

LaValle SM (2006) *Planning algorithms*. Cambridge university press.

LaValle SM and Kuffner JJ (2001) Randomized kinodynamic planning. *The Int. J. Robot. Res.* 20(5): 378–400.

Levine DS (2010) *Information-rich path planning under general constraints using rapidly-exploring random trees*. Master's Thesis, Massachusetts Institute of Technology.

Lin J (1991) Divergence measures based on the Shannon entropy. *IEEE Trans. Information Theory* 37(1): 145–151.

Makarenko A, Williams S, Bourgault F and Durrant-Whyte H (2002) An experiment in integrated exploration. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, volume 1. pp. 534–539.

Marchant R and Ramos F (2012) Bayesian optimisation for intelligent environmental monitoring. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, pp. 2242–2249.

Melkumyan A and Ramos F (2009) A sparse covariance function for exact Gaussian process inference in large datasets. In: *IJCAI*, volume 9. pp. 1936–1942.

Minka TP (2001) *A family of algorithms for approximate Bayesian inference*. PhD Thesis, Massachusetts Institute of Technology.

Montemerlo M, Thrun S, Koller D and Wegbreit B (2003) FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In: *In Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*. pp. 1151–1156.

Montemerlo M, Thrun S, Koller D, Wegbreit B et al. (2002) Fastslam: A factored solution to the simultaneous localization and mapping problem. In: *AAAI/IAAI*. pp. 593–598.

Moravec HP and Elfes A (1985) High resolution maps from wide angle sonar. In: *Proc. IEEE Int. Conf. Robot Automat.*, volume 2. IEEE, pp. 116–121.

Mount DM and Arya S (2006) ANN: A library for approximate nearest neighbor searching. URL `http://www.cs.umd.edu/~mount/ANN/`. Version 1.1.1.

Murphy KP (2012) *Machine learning: a probabilistic perspective*. The MIT Press.

Neal RM (1996) *Bayesian learning for neural networks*, volume 118. Springer New York.

Neumann M, Marthaler D, Huang S and Kersting K (2014) pyGPs; a library for Gaussian process regression and classification. https://github.com/marionmari/pyGPs.

Newman P and Ho K (2005) SLAM-loop closing with visually salient features. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 635–642.

O'Callaghan S and Ramos F (2011) Continuous occupancy mapping with integral kernels. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. pp. 1494–1500.

O'Callaghan S, Ramos FT and Durrant-Whyte H (2009) Contextual occupancy maps using Gaussian processes. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 1054–1060.

O'Callaghan ST, Ramos FT and Durrant-Whyte H (2010) Contextual occupancy maps incorporating sensor and location uncertainty. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 3478–3485.

Olson EB (2009) Real-time correlative scan matching. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 4387–4393.

Paz LM, Tardós JD and Neira J (2008) Divide and conquer: EKF SLAM in O(n). *IEEE Trans. Robot.* 24(5): 1107–1120.

Pineau J, Gordon G, Thrun S et al. (2003) Point-based value iteration: An anytime algorithm for POMDPs. In: *IJCAI*, volume 3. pp. 1025–1032.

Platt Jr R, Tedrake R, Kaelbling L and Lozano-Perez T (2010) Belief space planning assuming maximum likelihood observations .

Prentice S and Roy N (2009) The belief roadmap: Efficient planning in belief space by factoring the covariance. *The Int. J. Robot. Res.* 28(11-12): 1448–1465.

Press WH, Teukolsky SA, Vetterling WT and Flannery BP (1996) *Numerical recipes in C*, volume 2. Cambridge university press Cambridge.

Prestes e Silva E, Engel P, Trevisan M and Idiart M (2002) Exploration method using harmonic functions. *Robot. Auton. Syst.* 40(1): 25–42.

Pukelsheim F (2006) *Optimal design of experiments*. Society for Industrial and Applied Mathematics.

Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, Wheeler R and Ng AY (2009) ROS: an open-source Robot Operating System. In: *ICRA workshop on open source software*, volume 3. p. 5.

Ramos F and Ott L (2015) Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent. In: *Robotics: Science and Systems*.

Rappaport TS (1996) *Wireless communications: principles and practice*, volume 2. Prentice Hall PTR New Jersey.

Rasmussen C and Williams C (2006) *Gaussian processes for machine learning*, volume 1. MIT press.

Roy N, Burgard W, Fox D and Thrun S (1999) Coastal navigation-mobile robot navigation with uncertainty in dynamic environments. In: *Proc. IEEE Int. Conf. Robot Automat.*, volume 1. IEEE, pp. 35–40.

Roy N, Gordon GJ and Thrun S (2005) Finding approximate POMDP solutions through belief compression. *J. Artif. Intell. Res.(JAIR)* 23: 1–40.

Russell SJ and Norvig P (2009) *Artificial intelligence: a modern approach*. 3 edition. Prentice Hall.

Shade R and Newman P (2011) Choosing where to go: Complete 3D exploration with stereo. In: *Proc. IEEE Int. Conf. Robot Automat.* pp. 2806–2811.

Shen S, Michael N and Kumar V (2012) Stochastic differential equation-based exploration algorithm for autonomous indoor 3d exploration with a micro-aerial vehicle. *The Int. J. Robot. Res.* 31(12): 1431–1444.

Sibley D, Mei C, Reid I and Newman P (2009) Adaptive relative bundle adjustment. In: *Robotics: Science and Systems*, volume 32. p. 33.

Sibley G, Matthies L and Sukhatme G (2008) A sliding window filter for incremental SLAM. In: *Unifying perspectives in computational and robot vision*. Springer, pp. 103–112.

Sim R and Little JJ (2006) Autonomous vision-based exploration and mapping using hybrid maps and rao-blackwellised particle filters. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, pp. 2082–2089.

Sim R and Roy N (2005) Global A-optimal robot exploration in SLAM. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 661–666.

Simmons R and Koenig S (1995) Probabilistic robot navigation in partially observable environments. In: *IJCAI*, volume 95. pp. 1080–1087.

Singh A, Krause A, Guestrin C and Kaiser WJ (2009) Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research* : 707–755.

Singh A, Ramos F, Whyte HD and Kaiser WJ (2010) Modeling and decision making in spatio-temporal processes for environmental surveillance. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 5490–5497.

Smallwood RD and Sondik EJ (1973) The optimal control of partially observable markov processes over a finite horizon. *Operations Research* 21(5): 1071–1088.

Smith A, Doucet A, de Freitas N and Gordon N (2013) *Sequential Monte Carlo methods in practice*. Springer Science & Business Media.

Snelson E and Ghahramani Z (2006) Sparse Gaussian processes using pseudo-inputs. In: *Advances in Neural Information Processing Systems*. pp. 1257–1264.

Snelson E, Rasmussen CE and Ghahramani Z (2004) Warped Gaussian processes. *Advances in neural information processing systems* 16: 337–344.

Spaan MT and Vlassis N (2005) Perseus: Randomized point-based value iteration for POMDPs. *Journal of artificial intelligence research* : 195–220.

Stachniss C, Grisetti G and Burgard W (2005) Information gain-based exploration using rao-blackwellized particle filters. In: *Robotics: Science and Systems*, volume 2.

Stein M (1999) *Interpolation of spatial data: some theory for kriging*. Springer Verlag.

Strasdat H, Montiel J and Davison AJ (2010) Real-time monocular SLAM: Why filter? In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 2657–2664.

Ström DP, Nenci F and Stachniss C (2015) Predictive exploration considering previously mapped environments. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 2761–2766.

Sucan IA, Moll M and Kavraki LE (2012) The open motion planning library. *IEEE Robotics & Automation Magazine* 19(4): 72–82.

Surmann H, Nüchter A and Hertzberg J (2003) An autonomous mobile robot with a 3d laser range finder for 3d exploration and digitalization of indoor environments. *Robot. Auton. Syst.* 45(3): 181–198.

T O'Callaghan S and Ramos F (2012) Gaussian process occupancy maps. *The Int. J. Robot. Res.* 31(1): 42–62.

Thrun S (2003a) Exploring artificial intelligence in the new millennium. chapter Robotic Mapping: A Survey. Morgan Kaufmann Publishers Inc. ISBN 1-55860-811-7, pp. 1–35.

Thrun S (2003b) Learning occupancy grid maps with forward sensor models. *Autonomous robots* 15(2): 111–127.

Thrun S, Burgard W and Fox D (2005) *Probabilistic robotics*, volume 1. MIT press.

Thrun S, Liu Y, Koller D, Ng AY, Ghahramani Z and Durrant-Whyte H (2004) Simultaneous localization and mapping with sparse extended information filters. *The Int. J. Robot. Res.* 23(7-8): 693–716.

Thrun S and Montemerlo M (2006) The graph SLAM algorithm with applications to large-scale mapping of urban structures. *The Int. J. Robot. Res.* 25(5-6): 403–429.

Titsias MK and Lázaro-Gredilla M (2011) Variational heteroscedastic Gaussian process regression. In: *Proc. Int. Conf. Machine learning*. ACM, pp. 841–848.

Tresp V (2000) A Bayesian committee machine. *Neural Computation* 12(11): 2719–2741.

Valencia R, Morta M, Andrade-Cetto J and Porta JM (2013) Planning reliable paths with Pose SLAM. *IEEE Trans. Robot.* 29(4): 1050–1059.

Valencia R, Valls Miro J, Dissanayake G and Andrade-Cetto J (2012) Active Pose SLAM. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* pp. 1885–1891.

Valiente D, Ghaffari Jadidi M, Valls Miro J, Gil A and Reinoso O (2015) Information-based view initialization in visual SLAM with a single omnidirectional camera. *Robot. Auton. Syst.* 72: 93 – 104.

Vallvé J and Andrade-Cetto J (2013) Mobile robot exploration with potential information fields. In: *6th European Conference on Mobile Robots*.

Vallvé J and Andrade-Cetto J (2014) Dense entropy decrease estimation for mobile robot exploration. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 6083–6089.

Vallvé J and Andrade-Cetto J (2015) Potential information fields for mobile robot exploration. *Robot. Auton. Syst.* 69: 68–79.

Van Den Berg J, Abbeel P and Goldberg K (2011) LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *The Int. J. Robot. Res.* 30(7): 895–913.

Vasudevan S, Ramos F, Nettleton E and Durrant-Whyte H (2009) Gaussian process modeling of large-scale terrain. *Journal of Field Robotics* 26(10): 812–840.

Von Mises R (1964) *Mathematical theory of probability and statistics.* Academic Press.

Wurm KM, Hornung A, Bennewitz M, Stachniss C and Burgard W (2010) OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In: *Proc. of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation*, volume 2.

Yamauchi B (1997) A frontier-based approach for autonomous exploration. In: *Int. Sym. Comput. Intell. Robot. Automat.* pp. 146–151.

Yamauchi B (1998) Frontier-based exploration using multiple robots. In: *Int. Conf. Autonomous Agents*. pp. 47–53.

Yang K, Keat Gan S and Sukkarieh S (2013) A Gaussian process-based RRT planner for the exploration of an unknown and cluttered environment with a UAV. *Advanced Robotics* 27(6): 431–443.

Yu J, Karaman S and Rus D (2015) Persistent monitoring of events with stochastic arrivals at multiple stations. *IEEE Trans. Robot.* 31(3): 521–535.

Zlot R, Stentz A, Dias M and Thayer S (2002) Multi-robot exploration controlled by a market economy. In: *Proc. IEEE Int. Conf. Robot Automat.*, volume 3. pp. 3016–3023.