

Gaussian Process Autonomous Mapping and Exploration for Range Sensing Mobile Robots^{*}

Maani Ghaffari Jadidi[†]

Jaime Valls Miro[‡]

Gamini Dissanayake[‡]

October 18, 2017

Abstract

Most of the existing robotic exploration schemes use 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 a Gaussian Processes (GPs) occupancy mapping technique that is computationally tractable for online map building due to its incremental formulation and provides 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. We also propose a mutual information-based greedy exploration technique built on that representation that takes into account all possible future observations. A major advantage of high-dimensional map inference is the fact that such techniques require fewer observations, leading to a faster map entropy reduction during exploration for map building scenarios. Evaluations using the publicly available datasets show the effectiveness of the proposed framework for robotic mapping and exploration tasks.

^{*}Accepted for Autonomous Robots. mani.ghaffari@gmail.com – <http://maanighaffari.com>

[†]College of Engineering, University of Michigan, Ann Arbor, MI 48109 USA

[‡]Centre for Autonomous Systems (CAS), University of Technology Sydney, NSW 2007, Australia

Contents

1	Introduction	3
1.1	Contributions	4
1.2	Notation	5
1.3	Outline	5
2	Related work	6
3	Mapping	7
3.1	Gaussian Processes	8
3.2	Problem statement and formulation	9
3.3	Sensor model, training and test data	9
3.4	Map management	11
3.5	I-GPOM2; an improved mapping strategy	13
3.6	Frontier map	15
3.7	Computational complexity	17
4	Exploration	17
4.1	Mutual information algorithm	18
4.2	Decision making	20
4.3	Map regeneration	21
5	Results and Discussion	21
5.1	Experimental setup	22
5.2	Exploration results in the Intel map	24
5.3	Outdoor scenario: Freiburg Campus	25
6	Conclusion and Future Work	28

1 Introduction

Exploring an unknown environment without any prior knowledge gives 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*. This leads the problem towards the sequential decision making under imperfect state information which is known to be NP-hard (Singh et al. 2009).

Autonomous mobile robots are required to generate a spatial representation of the robot environment, this is known as the mapping problem. Solving this problem is an integral part of all autonomous navigation systems as the map encapsulates the knowledge of the robot about its surrounding. In robotic navigation tasks, a representation (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. This is known as the autonomous exploration problem in robotics. In this article, we are concerned with autonomous exploration for map building when the robot pose is estimated by an appropriate strategy such as Pose SLAM (Ila et al. 2010).

We develop a Gaussian Processes (GPs) occupancy mapping algorithm that is tailored for robotic navigation and is computationally tractable due to its incremental formulation. This representation has been shown to be superior to the traditional occupancy grid map (Moravec and Elfes 1985; Elfes 1987) as it captures structural correlations present in the environment and produces a continuous representation of the sensing uncertainty in the map space (O’Callaghan et al. 2009; T O’Callaghan and Ramos 2012; Kim and Kim 2012, 2013a,b; Ghaffari Jadidi et al. 2013a,b, 2014, 2015; Kim and Kim 2015; Ghaffari Jadidi et al. 2017b). Furthermore, this representation has applications beyond the occupancy mapping problem and it has been applied for large-scale terrain modeling (Vasudevan et al. 2009), active learning (Krause and Guestrin 2007), building maps to predict the fluid concentration (Stachniss et al. 2008), informative path planning (Binney and Sukhatme 2012), robotic information gathering (Hollinger and Sukhatme 2014; Hollinger 2015; Ghaffari Jadidi et al. 2016), and high-dimensional semantic map representation (Ghaffari Jadidi et al. 2017a).

In the problem of robotic exploration for map building, targets are usually defined by geometric *frontiers* extracted from the Occupancy Grid Map (OGM) (Yamauchi 1997). We propose an algorithm to extract frontiers from Gaussian Processes Occupancy Maps (GPOMs) representations in the form of a probability map. Furthermore, we develop an algorithm to numerically calculate the Mutual Information (MI) between the map and future measurements on that representation. MI is a measure of the value of information that quantifies the information gain from sensor measurements (Krause and Guestrin 2005). The maximum expected utility 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. The proposed MI algorithm takes into account all possible future measurements (by taking expectations over them), and therefore, is a suitable utility function.

The MI-based utility function is computed at the centroids of geometric frontiers and the frontier with

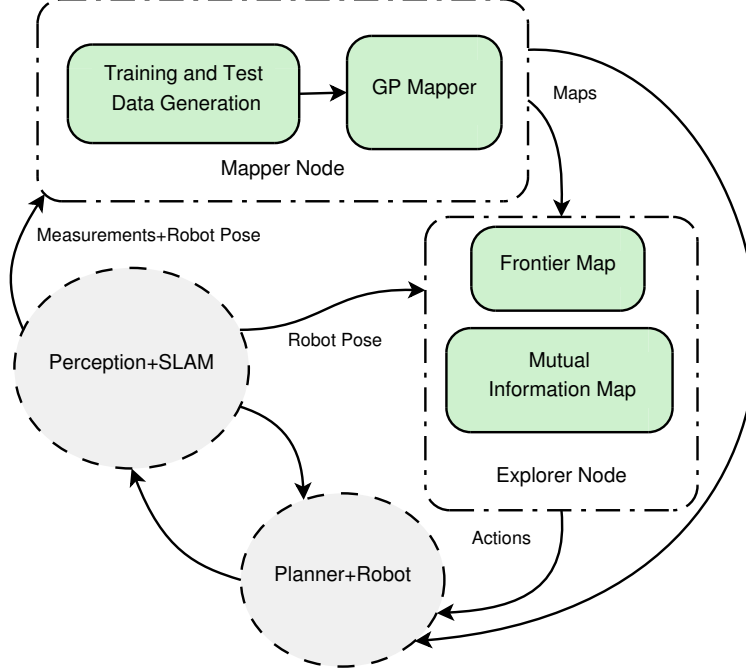


Figure 1: Schematic illustration of the autonomous mapping and exploration process using GPs maps. The GP mapper module provides the continuous occupancy map which can be exploited to extract geometric frontiers and mutual information maps. The maps also give support to the planner module for basic navigation 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.

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

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. Figure 1 depicts the proposed mutual information-based navigation process concept using GPOMs.

1.1 Contributions

This article is based on our preliminary work (Ghaffari Jadidi et al. 2015) where we presented continuous probabilistic frontier maps and an algorithm to calculate MI between the current map and future measurements. However, the distinguishable items from the previously published work are as follows:

- We expand the decision making part and use the notion of macro-action.
- We provide details about the derivation of the sensor model and the construction of the training and test point sets.

- We present more detailed evaluations with comparable techniques.

The main contributions of this work are as follows.

- A framework for incremental Gaussian processes occupancy mapping using range-finder sensors is developed. The method runs significantly faster with performance close to batch computation.
- We develop a novel probabilistic geometric frontier representation by exploiting continuous occupancy mapping and using L^1 -norm of the gradient of continuous occupancy maps.
- A tractable technique for greedy information gain-based exploration is developed that takes into account all possible future observations. The technique can deal with sparse measurements and uses a forward sensor model for map predictions.
- The results from publicly available datasets in a highly structured indoor environment and a large-scale outdoor space are presented.

1.2 Notation

In the present article probabilities and probability densities are not distinguished in general. Matrices are capitalized in bold, such as in \mathbf{X} , and vectors are in lower case bold type, such as in \mathbf{x} . Vectors are column-wise and $1:n$ means integers from 1 to n . The Euclidean norm is shown by $\|\cdot\|$. $|\mathbf{X}|$ denotes the determinant of matrix \mathbf{X} . 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, and the cardinality of the set is denoted by $|\mathcal{X}|$. A subscript asterisk, such as in \mathbf{x}_* , indicates a reference to a test set quantity. The n -by- n identity matrix is denoted by \mathbf{I}_n . $\text{vec}(x^{[1]}, \dots, x^{[n]})$ denotes a vector such as \mathbf{x} constructed by stacking $x^{[i]}$, $\forall i \in \{1:n\}$. The function notation is overloaded based on the output type and denoted by $k(\cdot)$, $\mathbf{k}(\cdot)$, and $\mathbf{K}(\cdot)$ where the outputs are scalar, vector, and matrix, 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.

1.3 Outline

The remaining parts of this article are organized as follows. In Section 2, a literature review is given. In Section 3, we present the proposed mapping algorithms and provide evaluations and comparison with occupancy grid maps. The exploration approach, MI surface calculation, and decision making process proposed in this work are discussed in Section 4. Section 5 presents the results from experiments in two difference scenarios. Finally, Section 6 concludes the article and discusses the limitations of this work.

2 Related work

An environment can be explored by directing a robot towards frontiers that indicate unknown regions of the environment in the neighboring known free areas (Yamauchi 1997). Traditional autonomous exploration strategies have been devised to use OGM (Moravec and Elfes 1985; Elfes 1987; Konolige 1997; Thrun 2003; Hornung et al. 2013; Merali and Barfoot 2014) to represent free, occupied and unknown regions. The following works use the concept of *active perception* (Bajcsy 1988) to take actions that reduce the uncertainty in the 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 while 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; a total utility function consisting of the weighted sum of the OGM entropy, navigation cost, and localizability is used. To enhance the map quality of the EKF-based Simultaneous Localization And Mapping (SLAM) (Cadena et al. 2016), an A-optimal criterion for autonomous exploration is examined in Sim and Roy (2005). Later in Carrillo et al. (2012), it is shown that the D-optimal (Determinant optimal) criterion (Pukelsheim 2006) is more effective in such scenarios.

In Stachniss et al. (2005), Rao-Blackwellized Particle Filters (RBPF) (Doucet et al. 2000) 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 Blanco et al. (2008), through the entropy of the expected map of RBPF, the technique takes the uncertainty in both robot path and map into account. In a similar framework Carlone et al. (2010, 2014) addressed the problem of active SLAM and exploration, specifically the inconsistency in the filter due to information loss for a given policy using the relative entropy concept. 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. Similar to this work, these techniques use 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 Carrillo et al. (2015), to avoid the need to update the map using unknown future measurements, the objective function is simplified to the current map entropy. 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. The technique in [Charrow et al. \(2015\)](#) is closely related to this work. The computational performance of the information gain is increased by using Cauchy-Schwarz Quadratic Mutual Information (CSQMI). It is shown that the behavior of CSQMI is similar to that of MI while it can be computed faster. The technique has also been further extended to the multi-robot scenario ([Faigl et al. 2012](#); [Charrow et al. 2014](#); [Charrow 2015](#)).

The methods reviewed above fall short of accounting for structural correlations in the environment. Kernel methods in the form of a Gaussian Processes framework ([Rasmussen and Williams 2006](#)) are non-parametric regression and classification techniques that have been extensively used by researchers 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 variance surface of GPs equate to a continuous representation of uncertainty in the environment, which it 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 ([Yang et al. 2013](#)). The incremental GP map building using the Bayesian Committee Machine (BCM) technique ([Tresp 2000](#)) is developed in [Kim and Kim \(2012\)](#); [Ghaffari Jadidi et al. \(2013a,b, 2014\)](#) and for online applications in [Wang and Englot \(2016\)](#). In [Ramos and Ott \(2015\)](#), the Hilbert maps technique is proposed that is more scalable and can be updated in linear time. However, it approximates the problem and produces maps with less accuracy than GPOM.

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 (see Section 3.7) is prohibitive for scenarios such as robotic navigation where a dense representation is preferred. The incremental GP map building was studied in [Kim and Kim \(2012\)](#), and in [Ghaffari Jadidi et al. \(2013a,b, 2014\)](#). In this 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.

3 Mapping

The GP mapper module is shown in Figure 2 which 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 frame through the BCM technique ([Tresp 2000](#)).

Before formal statement of the problem, we clarify the following assumptions.

Assumption 1 (Static environment). The environment that the robot navigates in is static.

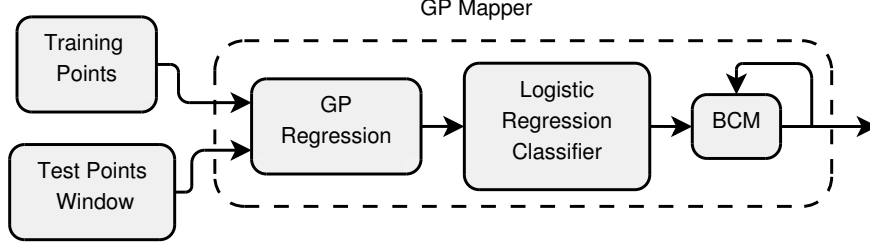


Figure 2: Schematic illustration of GP Mapper module. GP models the correlation in data and place distributions on test points. The logistic regression classifier squashes the output of GP into probabilities and returns the local map where the BCM module updates the global map incrementally.

Assumption 2 (Gaussian occupancy map points). Any sampled point from the occupancy map representation of the environment is a random variable whose distribution is Gaussian.

3.1 Gaussian Processes

A Gaussian Process is a collection of any finite number of random variables which are jointly distributed Gaussian (Rasmussen and Williams 2006). The joint distribution of the observed target values, \mathbf{y} , and the function values (the latent variable), \mathbf{f}_* , at the query points can be written as

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, \begin{bmatrix} \mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_n & \mathbf{K}(\mathbf{X}, \mathbf{X}_*) \\ \mathbf{K}(\mathbf{X}_*, \mathbf{X}) & \mathbf{K}(\mathbf{X}_*, \mathbf{X}_*) \end{bmatrix}) \quad (1)$$

where \mathbf{X} is the $d \times n$ design matrix of aggregated input vectors \mathbf{x} , \mathbf{X}_* is a $d \times n_*$ query points matrix, $\mathbf{K}(\cdot, \cdot)$ is the GP covariance matrix, and σ_n^2 is the variance of the observation noise which is assumed to have an independent and identically distributed (i.i.d.) Gaussian distribution. Define a training set $\mathcal{D} = \{(\mathbf{x}^{[i]}, y^{[i]}) \mid i = 1 : n\}$. The predictive conditional distribution for a single query point $f_* | \mathcal{D}, \mathbf{x}_* \sim \mathcal{N}(\mathbb{E}[f_*], \mathbb{V}[f_*])$ can be derived as

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

$$\sigma = \mathbb{V}[f_*] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}(\mathbf{X}, \mathbf{x}_*)^T [\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_n]^{-1} \mathbf{k}(\mathbf{X}, \mathbf{x}_*) \quad (3)$$

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) and hereby we select them as the kernel of GPs. For a single query point \mathbf{x}_* the function is given by

$$k(\mathbf{x}, \mathbf{x}_*) = \frac{1}{\Gamma(\nu) 2^{\nu-1}} \left[\frac{\sqrt{2\nu} \|\mathbf{x} - \mathbf{x}_*\|}{\kappa} \right]^\nu K_\nu \left(\frac{\sqrt{2\nu} \|\mathbf{x} - \mathbf{x}_*\|}{\kappa} \right) \quad (4)$$

where Γ is the Gamma function, $K_\nu(\cdot)$ is the modified Bessel function of the second kind of order ν , κ is the

characteristic length scale, and ν is a positive parameter used to control the smoothness of the covariance.

The hyperparameters of the covariance and mean function, θ , can be computed by minimization of the negative log of the marginal likelihood (NLML) function.

$$\log p(\mathbf{y}|\mathbf{X}, \theta) = -\frac{1}{2}\mathbf{y}^T [\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_n]^{-1} \mathbf{y} - \frac{1}{2} \log |\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}_n| - \frac{n}{2} \log 2\pi \quad (5)$$

3.2 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 $(M^{[1]}, \dots, M^{[n_m]})$ 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 $(Z^{[1]}, \dots, Z^{[n_z]})$ whose elements can take values $z^{[k]} \in \mathcal{Z}$, $k \in \{1 : n_z\}$. Let $\mathcal{X} \subset \mathbb{R}^2$ be the set of spatial coordinates to build a map on. Let $\mathbf{x}_o^{[k]} \in \mathcal{X}_o \subset \mathcal{X}$ be an observed occupied point by the k -th sensor beam from the environment which, at any time-step t , can be calculated by transforming the local observation $\mathbf{z}^{[k]}$ to the global frame using the robot pose $\mathbf{x}_t \in \text{SE}(2)$. Let $\mathbf{X}_f^{[k]} \in \mathcal{X}_f \subset \mathcal{X}$ 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 $\mathcal{D}_o = \{(\mathbf{x}_o^{[i]}, \mathbf{y}_o^{[i]}) \mid i = 1 : n_o\}$ and a training set of unoccupied points $\mathcal{D}_f = \{(\mathbf{x}_f^{[i]}, \mathbf{y}_f^{[i]}) \mid i = 1 : n_f\}$ in which $\mathbf{y}_o = \text{vec}(\mathbf{y}_o^{[1]}, \dots, \mathbf{y}_o^{[n_o]})$ and $\mathbf{y}_f = \text{vec}(\mathbf{y}_f^{[1]}, \dots, \mathbf{y}_f^{[n_f]})$ 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 \mathbf{x}_t and observations $Z_t = \mathbf{z}_t$, we wish to estimate $p(M = m \mid \mathbf{x}_t, Z_t = \mathbf{z}_t)$. Place a joint distribution over M ; the map can be inferred as a Gaussian process by defining the process as the function $y : \mathcal{X} \rightarrow \mathcal{M}$, therefore

$$y(\mathbf{x}) \sim \mathcal{GP}(f_m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')) \quad (6)$$

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

$$m^{[i]} = y(\mathbf{x}_*)^{[i]} \sim \mathcal{N}(\mu^{[i]}, \sigma^{[i]}) \quad (7)$$

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), (Ghaffari Jadidi et al. 2014), squashes data into the range $[0, 1]$.

3.3 Sensor model, training and test 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

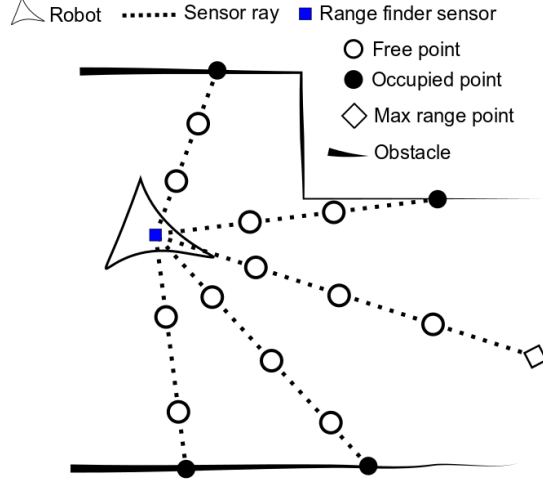


Figure 3: Conceptual illustration of the robot, the environment, and observations. Training data consists of free and occupied points labeled $y_f = -1$ and $y_o = +1$ respectively. Free points are sampled along each beam, i.e. negative sensor information while occupied points are directly observable.

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 3 shows the conceptual illustration of the environment and training points generation.

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

$$\mathbf{z}_t^{[k]} = \begin{bmatrix} r_t^{[k]} \\ \alpha_t^{[k]} \end{bmatrix} = h(\mathbf{x}_t, \mathbf{x}_o^{[k]}) + \mathbf{v}, \quad \mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \quad (8)$$

$$h(\mathbf{x}_t, \mathbf{x}_o^{[k]}) \triangleq \begin{bmatrix} \sqrt{(\mathbf{x}_o^{[k,1]} - \mathbf{x}_t^{[1]})^2 + (\mathbf{x}_o^{[k,2]} - \mathbf{x}_t^{[2]})^2} \\ \arctan(\mathbf{x}_o^{[k,2]} - \mathbf{x}_t^{[2]}, \mathbf{x}_o^{[k,1]} - \mathbf{x}_t^{[1]}) - \mathbf{x}_t^{[3]} \end{bmatrix} \quad (9)$$

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 \mathbf{v} is assumed to be Gaussian with zero mean and covariance \mathbf{R} . To find $\mathbf{x}_o^{[k]}$ which is in the map space, the inverse model can be calculated as

$$\mathbf{x}_o^{[k]} = \mathbf{x}_t^{[1:2]} + r_t^{[k]} R(\mathbf{x}_t^{[3]}) \begin{bmatrix} \cos(\alpha_t^{[k]}) \\ \sin(\alpha_t^{[k]}) \end{bmatrix} \quad (10)$$

where $R(\mathbf{x}_t^{[3]}) \in \text{SO}(2)$ 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 $\mathcal{D}_o = \{(\mathbf{x}_o^{[k]}, y_o^{[k]}) \mid k = 1 : n_z\}$. 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 $\mathbf{x}_o^{[k]}$ as its end points. Therefore,

$$\mathbf{X}_f^{[k,j]} = \mathbf{x}_t^{[1:2]} + \delta_j R(\mathbf{x}_t^{[3]}) \begin{bmatrix} \cos(\alpha_t^{[k]}) \\ \sin(\alpha_t^{[k]}) \end{bmatrix} \quad (11)$$

where $\delta_j \sim \mathcal{U}(0, r_t^{[k]})$ $j = 1 : n_f^{[k]}$, $\mathcal{U}(0, r_t^{[k]})$ 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_{fmin} . Therefore we can write

$$n_f^{[k]} \triangleq \max(\{n_{fmin}, s_l(r_t^{[k]})\}) \quad (12)$$

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 cubically. We can construct the training set of unoccupied points as $\mathcal{D}_f = \bigcup_{i=1}^{n_z} \mathcal{D}_f^{[i]}$ where $\mathcal{D}_f^{[i]} = \{(\mathbf{X}_f^{[k]}, \mathbf{y}_f^{[k]}) \mid k = 1 : n_z\}$ and $\mathbf{y}_f^{[k]} = \text{vec}(\mathbf{y}_f^{[1]}, \dots, \mathbf{y}_f^{[n_f^{[k]}]})$.

Remark 1. Generally speaking, query points can have any desired distributions and the actual representation of the map depends on that distribution. However, building the map over a grid facilitates comparison with standard occupancy grid-based methods, i.e. at similar map resolutions. We use function `TestDataWindow`, in Algorithms 1 and 4, for generating a grid at a given position. The size of this grid can be set according to the maximum sensor range, the environment size, or available computational resources for data processing.

Remark 2. Throughout all algorithms, when we write m for a map, it is assumed that the mean μ , the variance σ , the occupancy probability $p(m)$, and the corresponding spatial coordinates are available even if they are not mentioned or used explicitly. For simplicity, when m is used for computations such as in $\log(p(m))$, we write $\log(m)$.

3.4 Map management

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, in order to keep track of the global map, a map management step is required where the local 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.

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

Algorithm 1 IGPOM()

Require: Robot pose \mathbf{p} and measurements \mathbf{z} ;

- 1: **if** firstFrame **then**
 - 2: $m \leftarrow \emptyset$ // Initialize the map
 - 3: optimize GP hyperparameters θ // Minimize the NLML, Equation (5)
 - 4: **end if**
 - 5: $\mathbf{X}_* \leftarrow \text{TestDataWindow}(\mathbf{p})$ // Query points grid centered at the robot pose
 - 6: $\mathbf{X}_o, \mathbf{y}_o \leftarrow \text{Transform2Global}(\mathbf{p}, \mathbf{z})$ // Occupied training data, label +1, Equation (10)
 - 7: $\mathbf{X}_f, \mathbf{y}_f \leftarrow \text{TrainingData}(\mathbf{p}, \mathbf{z})$ // Unoccupied training data, label -1, Equation (11)
 - 8: $[\mu_*, \sigma_*] \leftarrow \text{GP}(\theta, [\mathbf{X}_o; \mathbf{X}_f], [\mathbf{y}_o; \mathbf{y}_f], \mathbf{X}_*)$ // Compute predictive mean and variance, Equation (2) and (3)
 - 9: $m \leftarrow \text{UpdateMap}(\mu_*, \sigma_*, m)$ // Algorithm 3
 - 10: **return** m
-

Algorithm 2 FusionBCM($\mu_a, \mu_b, \sigma_a, \sigma_b$)

- 1: $\sigma_c \leftarrow (\sigma_a^{-1} + \sigma_b^{-1})^{-1}$ // Point-wise calculation of Equation (14)
 - 2: $\mu_c \leftarrow \sigma_c(\sigma_a^{-1}\mu_a + \sigma_b^{-1}\mu_b)$ // Point-wise calculation of Equation (13)
 - 3: **return** μ_c, σ_c
-

and covariance Σ and each GP with mean $\mathbb{E}[f_*|\mathcal{D}^{[i]}]$ and covariance $\mathbb{C}_{\odot\mathbb{V}}[f_*|\mathcal{D}^{[i]}]$, it follows that (Tresp 2000)

$$\mathbb{E}[f_*|\mathcal{D}] = \mathbf{C}^{-1} \sum_{i=1}^{p_m} \mathbb{C}_{\odot\mathbb{V}}[f_*|\mathcal{D}^{[i]}]^{-1} \mathbb{E}[f_*|\mathcal{D}^{[i]}] \quad (13)$$

$$\mathbf{C} = \mathbb{C}_{\odot\mathbb{V}}[f_*|\mathcal{D}]^{-1} = -(p_m - 1)(\Sigma)^{-1} + \sum_{i=1}^{p_m} \mathbb{C}_{\odot\mathbb{V}}[f_*|\mathcal{D}^{[i]}]^{-1} \quad (14)$$

where p_m is the total number of mapping processes. In this work, we use BCM for combining a local and a previously existing global map, or merging two global maps; therefore $p_m = 2$. In addition, in the case of uninformative prior over map points the term Σ^{-1} can be set to zero, i.e. very large covariances/variances.

The steps of the incremental GPOM (I-GPOM) are shown in Figure 2 and Algorithms 1, 2, and 3, where a BCM module updates the global map as new observations are taken. In Figure 4 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. 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, a set of 25 laser scans, where each scan contains about 180 points, 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 to 29. The proposed incremental mapping approach using BCM performs accurate and close to the batch form even with about 8 steps intermission between successive observations and is faster.

Algorithm 3 UpdateMap()

Require: Global map m , μ , σ and local map m_* , μ_* , σ_* ;

- 1: **for** all $i \in \mathcal{M}_*$ **do**
 - 2: $j \leftarrow$ find the corresponding global index of i using the map spatial coordinates and a nearest neighbor search
 - 3: $\mu^{[j]}, \sigma^{[j]} \leftarrow \text{FusionBCM}(\mu^{[j]}, \mu_*^{[i]}, \sigma^{[j]}, \sigma_*^{[i]})$ // Algorithm 2
 - 4: **end for**
 - 5: $m \leftarrow \text{LogisticRegression}(\mu, \sigma)$ // Squash data into (0,1)
 - 6: **return** m
-

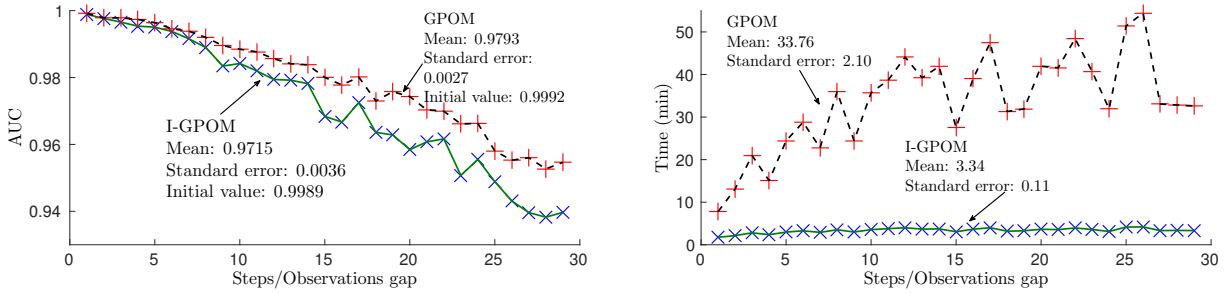


Figure 4: Comparison of I-GPOM and batch GPOM methods using the Intel dataset with the observations size of 25 laser scans at each step due to the memory limitation for the batch GP computations. The left plot shows the AUC and the right plot depicts the runtime for each step. The horizontal axes indicate observations gaps. As the number of gaps grows, the batch GP outperforms the incremental method as it learns the correlation between observations at once; however, with higher computational time. On the other hand, the incremental method in nearly constant time per update produces a similar average map quality with the mean difference of 0.0078.

Optimization of the hyper-parameters is performed once at the beginning of each experiment by minimization of the negative log of the marginal likelihood function. For the prevailing case of multiple runs in the same environment, the optimized values can then be loaded off-line.

3.5 I-GPOM2; an improved mapping strategy

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)).

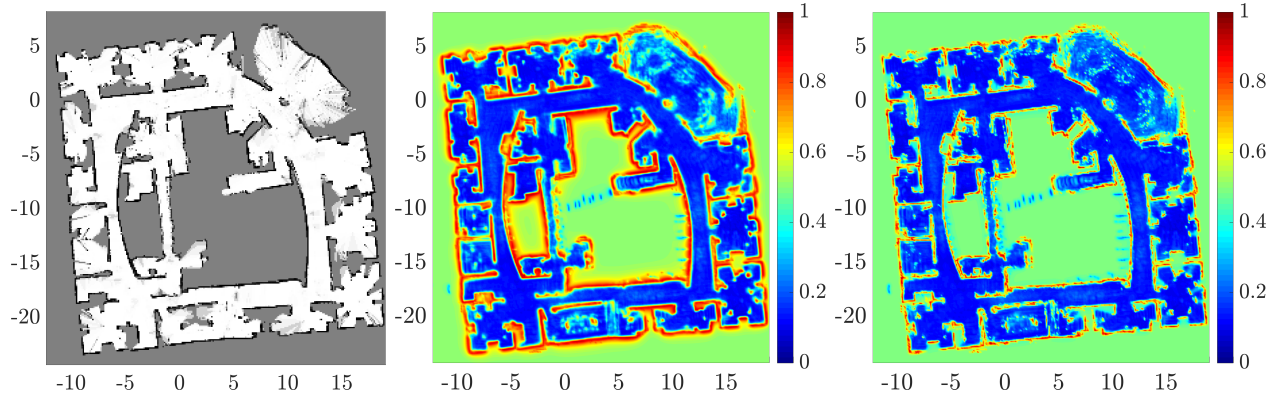


Figure 5: Occupancy maps visualization; from left to right: OGM, I-GPOM, I-GPOM2. The maps are build incrementally using all observations available in the Intel dataset. For the I-GPOM and I-GPOM2 maps the Matérn ($\nu = 3/2$) covariance function is used. I-GPOM and I-GPOM2 can complete partially observable 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.

Table 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

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 approach (O’Callaghan and Ramos 2011) can mitigate the first aforementioned deficiency, however, the integration over GPs kernels is computationally demanding and results in less tractable methods. In order 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 continuous occupancy map (I-GPOM2). The complete results of occupancy mapping with the three different methods in the Intel dataset are presented in Figure 5, while the AUCs are compared in Table 1. The I-GPOM2 method demonstrates more flexibility to model the 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 5 and is not reflected in the AUC of I-GPOM and I-GPOM2. Algorithms 4 and 5 encapsulate the I-GPOM2 methods as implemented in the present work.

Algorithm 4 IGPOM2()

Require: Robot pose \mathbf{p} and measurements \mathbf{z} ;

- 1: **if** firstFrame **then**
 - 2: $m, m_o, m_f \leftarrow \emptyset$ // Initialize the map
 - 3: optimize GP hyperparameters θ_o, θ_f // Minimize the NLML, Equation (5)
 - 4: **end if**
 - 5: $\mathbf{X}_* \leftarrow \text{TestDataWindow}(\mathbf{p})$ // Query points grid centered at the robot pose
 - 6: $\mathbf{X}_o, \mathbf{y}_o \leftarrow \text{Transform2Global}(\mathbf{p}, \mathbf{z})$ // Occupied training data, label +1, Equation (10)
 - 7: $\mathbf{X}_f, \mathbf{y}_f \leftarrow \text{TrainingData}(\mathbf{p}, \mathbf{z})$ // Unoccupied training data, label -1, Equation (11)
 - 8: $[\mu_{o*}, \sigma_{o*}] \leftarrow \text{GP}(\theta_o, \mathbf{X}_o, \mathbf{y}_o, \mathbf{X}_*)$ // Compute occupied map predictive mean and variance, Equation (2) and (3)
 - 9: $[\mu_{f*}, \sigma_{f*}] \leftarrow \text{GP}(\theta_f, \mathbf{X}_f, \mathbf{y}_f, \mathbf{X}_*)$ // Compute unoccupied map predictive mean and variance using (2) and (3)
 - 10: $m_o \leftarrow \text{UpdateMap}(\mu_{o*}, \sigma_{o*}, m_o)$ // Algorithm 3
 - 11: $m_f \leftarrow \text{UpdateMap}(\mu_{f*}, \sigma_{f*}, m_f)$
 - 12: $m \leftarrow \text{MergeMap}(m_o, m_f)$ // Algorithm 5
 - 13: **return** m, m_o
-

Algorithm 5 MergeMap()

Require: Unoccupied map m_f, μ_f, σ_f and occupied map m_o, μ_o, σ_o ;

- 1: **for** all $i \in \mathcal{M}$ **do**
 - 2: $\mu^{[i]}, \sigma^{[i]} \leftarrow \text{FusionBCM}(\mu_o^{[i]}, \mu_f^{[i]}, \sigma_o^{[i]}, \sigma_f^{[i]})$ // Algorithm 2
 - 3: **end for**
 - 4: $m \leftarrow \text{LogisticRegression}(\mu, \sigma)$ // Squash data into (0,1)
 - 5: **return** m
-

3.6 Frontier map

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, defining frontiers in a probabilistic form using map uncertainty is more suitable for computing expected behaviors. The boundaries that correspond to frontiers can be computed using the following heuristic formula.

$$\tilde{f}^{[i]} \triangleq \|\nabla p(m^{[i]})\|_1 - \beta(\|\nabla p(m_o^{[i]})\|_1 + p(m_o^{[i]}) - 0.5) \quad (15)$$

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 while $\|\nabla p(m_o^{[i]})\|_1$ 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 map uncertainty. To squash the frontier and variance values into the range $[0, 1]$, a logistic regression classifier

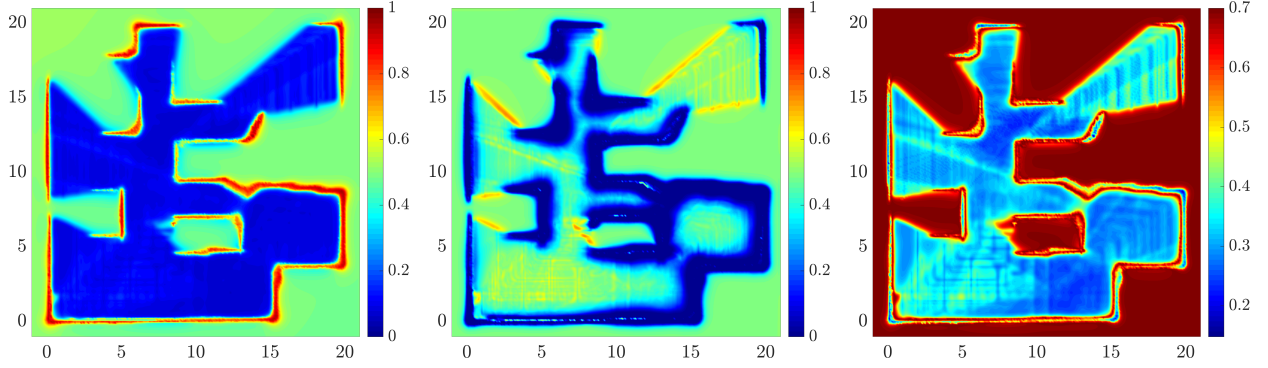


Figure 6: Inferred continuous occupancy map (left); associated probabilistic frontier map (middle); and mutual information surface (right, discussed in Section 4.1). The frontier map highlights the informative regions for further exploration by assigning higher probabilities to frontier points. The lower probabilities show the obstacles and walls while the values greater than the *no discrimination* probability, 0.5, can be considered as frontiers. In the MI surface, the areas beyond the current perception field of the robot preserve their initial entropy values and the higher values demonstrate regions with greater information gain. The map dimensions are in meters and the MI values in nats.

with inputs from $\bar{f}^{[i]}$ and map uncertainty $\sigma^{[i]}$ is applied to data which yields

$$p(f^{[i]}|m^{[i]}, w_f^{[i]}) = \frac{1}{1 + \exp(-w_f^{[i]} \bar{f}^{[i]})} \quad (16)$$

where $w_f^{[i]} = \gamma_f \sqrt{\lambda^{[i]}}$ denotes the required weights, $\lambda^{[i]} \triangleq \sigma_{min}/\sigma^{[i]}$ is the bounded information associated with location i , 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 6 (middle) depicts an instance of the frontier map from an exploration experiment in the Cave environment (Howard and Roy 2003).

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.
2. The binary map of frontiers is clustered into subsets of candidate macro-actions.
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 until the action set \mathcal{A}_t is not empty (repeats from step 1).

Algorithm 6 BuildFrontierMap()

Require: Current map m , σ and occupied map m_o , σ_o ;

```
1: // Compute boundaries
2:  $dm \leftarrow \|\nabla p(m)\|_1$ ,  $dm_o \leftarrow \|\nabla p(m_o)\|_1$ 
3:  $\sigma_{min} \leftarrow \min(\sigma)$ 
4:  $f \leftarrow \emptyset$ 
5: // Compute probabilistic frontiers
6: for all  $i \in \mathcal{M}$  do
7:    $\tilde{f}^{[i]} \leftarrow dm^{[i]} - \beta(dm_o^{[i]} + m_o^{[i]} - 0.5)$ 
8:    $w_f^{[i]} \leftarrow \gamma_f \text{sqrt}(\sigma_{min}/\sigma^{[i]})$  // Logistic regression weights
9:    $f^{[i]} \leftarrow (1 + \exp(-w_f^{[i]} \tilde{f}^{[i]}))^{-1}$  // Squash data into (0,1), Equation (16)
10: end for
11: return  $f$ 
```

3.7 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, $n_t = n_o + n_f$. 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 $\mathcal{O}(n_t^3 + n_q \log n_q + n_m)$.

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.

4 Exploration

In the context of autonomous robotic mapping, typically, the main goal is map completion while maintaining the localization accuracy at a reasonable level¹. Let a_t be an action from the set of all possible actions \mathcal{A}_t at time t . The goal is to choose the action that optimizes the desired objective function. In the following, we define the most common utility functions for the single robot exploration case.

Nearest Frontier. The nearest frontier policy drives the robot towards the closest frontier to its current pose. Geometric frontiers can be extracted from the occupancy map (Keidar and Kaminka 2013). For the GPOM technique, we use the probabilistic frontier map. 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, $f_c : \mathcal{A}_t \rightarrow \mathbb{R}_{\geq 0}$, is the length of the path from the current robot pose to the corresponding frontier.

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

Therefore,

$$a_t^* = \underset{a_t \in \mathcal{A}_t}{\operatorname{argmin}} f_c(a_t) \quad (17)$$

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.

Remark 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 (17) the integrand is simply 1.

Information Gain. Let $f_I(a_t)$, $f_I : \mathcal{A}_t \rightarrow \mathbb{R}_{\geq 0}$, be a function that quantifies the information quality of action a_t . To find the action that maximizes the information gain-based utility function, the problem can be written as

$$a_t^* = \underset{a_t \in \mathcal{A}_t}{\operatorname{argmax}} f_I(a_t) \quad (18)$$

In other words, the robot takes the action that leads to the maximum return of information. However, as it is evident from (18) the cost of taking that action is not included in the utility function.

Cost-Utility Trade-off. The third approach 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 (17) and (18). The primary problem is that the units of utility/cost functions are different. One solution is to express the cost in the form of information loss (uncertainty). Another approach is to combine them using appropriate coefficients, e.g. a linear combination of the utility and cost functions. Let $g : \mathbb{R}_{\geq 0}^2 \rightarrow \mathbb{R}_{\geq 0}$ 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.

$$a_t^* = \underset{a_t \in \mathcal{A}_t}{\operatorname{argmax}} u(a_t) \quad (19)$$

4.1 Mutual information algorithm

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 $Z = \mathbf{z}$ from \mathcal{Z} what will be the reduction in the map $M = m$ uncertainty? The MI between the map and the future measurement $Z_{t+1} = \hat{\mathbf{z}}$ is

$$\begin{aligned} I(M; Z_{t+1} | \mathbf{z}_{1:t}) &= \int_{\hat{\mathbf{z}} \in \mathcal{Z}} \sum_{m \in \mathcal{M}} p(m, \hat{\mathbf{z}} | \mathbf{z}_{1:t}) \log \frac{p(m, \hat{\mathbf{z}} | \mathbf{z}_{1:t})}{p(m | \mathbf{z}_{1:t}) p(\hat{\mathbf{z}} | \mathbf{z}_{1:t})} d\hat{\mathbf{z}} \\ &= H(M | \mathbf{z}_{1:t}) - \overline{H}(M | Z_{t+1}, \mathbf{z}_{1:t}) \end{aligned} \quad (20)$$

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

$$H(M|\mathbf{z}_{1:t}) = - \sum_{m \in \mathcal{M}} p(m|\mathbf{z}_{1:t}) \log p(m|\mathbf{z}_{1:t}) \quad (21)$$

$$\bar{H}(M|Z_{t+1}, \mathbf{z}_{1:t}) = \int_{\hat{\mathbf{z}} \in \mathcal{Z}} p(\hat{\mathbf{z}}|\mathbf{z}_{1:t}) H(M|Z_{t+1} = \hat{\mathbf{z}}, \mathbf{z}_{1:t}) d\hat{\mathbf{z}} \quad (22)$$

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

$$p(m^{[i]}|\hat{\mathbf{z}}_{t+1}^{[k]}, \mathbf{z}_{1:t}) = \frac{p(\hat{\mathbf{z}}_{t+1}^{[k]}|m^{[i]})p(m^{[i]}|\mathbf{z}_{1:t})}{p(\hat{\mathbf{z}}_{t+1}^{[k]}|\mathbf{z}_{1:t})} \quad (23)$$

$$p(\hat{\mathbf{z}}_{t+1}^{[k]}|\mathbf{z}_{1:t}) = \sum_{m^{[i]} \in \mathcal{M}} p(\hat{\mathbf{z}}_{t+1}^{[k]}|m^{[i]})p(m^{[i]}|\mathbf{z}_{1:t}) \quad (24)$$

The likelihood function $p(\hat{\mathbf{z}}_{t+1}^{[k]}|M = m^{[i]})$ is a beam-based mixture measurement model, where the term $p(\hat{\mathbf{z}}_{t+1}^{[k]}|M = 0)$ can be interpreted as the likelihood of not observing the map point at location i , i.e. uniform distribution. The term $p(\hat{\mathbf{z}}_{t+1}^{[k]}|\mathbf{z}_{1:t})$ is the marginal distribution over measurements which is denoted by p_z in line 18 of Algorithm 7. By numerically integrating over a desired beam range, we can compute the predicted map posterior entropy using Equation (22). 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

$$I^{[i]} = h(m^{[i]}) - \bar{h}(m^{[i]}) \quad (25)$$

where $h(m^{[i]})$ is the current entropy of the map point $m^{[i]}$ and $\bar{h}(m^{[i]})$ is the estimated map conditional entropy. In practice, at each time-step, the map is initialized with the current map entropy, $H(M|\mathbf{z}_{1:t})$, 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 6, an estimated MI map during an exploration experiments in the Cave environment (Howard and Roy 2003) is depicted.

Computational Complexity. For MI surface, the time complexity is at worst quadratic in the number of map points in the current perception field of the robot, $n_p = |\bigcup_{k=1}^{n_z} \mathcal{I}_{t+1}^{[k]}|$, and linear in the number of sensor beams, n_z , and numerical integration's resolution, s_z , resulting in $\mathcal{O}(n_p^2 n_z s_z)$.

Algorithm 7 BuildMIMap()

Require: Robot pose or desired location, current map estimate m , numerical integration resolution s_z , sensor model;

```
1:  $\bar{m} \leftarrow m$ 
2: // Initialize MI map using current map entropy
3:  $I \leftarrow -(m \log(m) + (1 - m) \log(1 - m))$ 
4: for all  $k$  do // Loop over all sensor beams
5:   Compute  $\hat{z}_{t+1}^{[k]}$  and  $\mathcal{I}_{t+1}^{[k]}$  using ray casting in  $m$ 
6:   // Calculate map conditional entropy along beam  $k$ 
7:   for  $i \in \mathcal{I}_{t+1}^{[k]}$  do
8:      $\bar{h} \leftarrow 0$  // Initialize map conditional entropy
9:      $z \leftarrow s_z^{-1}$  // Initialize range dummy variable
10:    while  $z \leq \hat{z}_{t+1}^{[k]}$  do
11:      // Calculate marginal measurement probability  $p_z$ , Equation (24)
12:       $p_1 \leftarrow p(z|M = 0)$ 
13:       $p_2 \leftarrow 0$ 
14:      for  $j \in \mathcal{I}_{t+1}^{[k]}$  do
15:         $p_1 \leftarrow p_1(1 - m^{[j]})$ 
16:         $p_2 \leftarrow p_2 +$ 
            $p(z|M = m^{[j]})m^{[j]} \prod_{l < j} (1 - m^{[l]})$ 
17:      end for
18:       $p_z \leftarrow p_1 + p_2$ 
19:      // Map prediction at point  $i$  along beam  $k$ 
20:       $\bar{m}^{[i]} \leftarrow p_z^{-1} p(z|M = m^{[i]}) m^{[i]} \prod_{l < i} (1 - m^{[l]})$ 
21:       $\bar{h} \leftarrow \bar{h} +$ 
            $p_z[\bar{m}^{[i]} \log(\bar{m}^{[i]}) + (1 - \bar{m}^{[i]}) \log(1 - \bar{m}^{[i]})]$ 
22:       $z \leftarrow z + s_z^{-1}$  // Increase range along the beam
23:    end while
24:     $I^{[i]} \leftarrow I^{[i]} + \bar{h}s_z^{-1}$  // Equation (25)
25:  end for
26: end for
27: return  $I$ 
```

4.2 Decision making

Let each geometric frontier be regarded as a macro-action. 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

$$f_I(a_t) \triangleq \sum_{k=1}^{n_z} \sum_{i \in \mathcal{I}^{[k]}} I^{[i]}(a_t) \quad (26)$$

$$u(a_t) \triangleq \alpha f_I(a_t) - f_c(a_t) \quad (27)$$

where α is a factor to relate information gain to the cost of motion. Note that the expectation over future measurements and path lengths is already incorporated into the information and cost functions.

The optimal action a_t^* 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. \(2003\)](#).

4.3 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, \dots, p_n , with weights $\pi_1, \pi_2, \dots, \pi_n$ is

$$JS_\pi(p_1, p_2, \dots, p_n) = H\left(\sum_{i=1}^n \pi_i p_i\right) - \sum_{i=1}^n \pi_i H(p_i) \quad (28)$$

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 4. The main advantage of JSD over Kullback-Leibler divergence, in this case, is that JSD is bounded. As a result, it is more suitable for decision making ([Lin 1991](#)).

5 Results and Discussion

We now 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.

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)



Figure 7: The constructed environment for exploration experiments using the binary map of obstacles from the Intel dataset.

Table 2: 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^*

(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 2.

5.1 Experimental setup

The environment is constructed using a binary map of obstacles and, for the Intel map, is shown in Figure 7. The simulated robot is equipped with odometric and laser range-finder sensors to provide the required sensory inputs for Pose SLAM. The odometric and laser range-finder sensors noise covariances are set to $\Sigma_u = \text{diag}(0.1 \text{ m}, 0.1 \text{ m}, 0.0026 \text{ rad})^2$ and $\Sigma_y = \text{diag}(0.03 \text{ m}, 0.03 \text{ m}, 0.0013 \text{ rad})^2$, respectively. The motion of the robot is modeled using a velocity motion model (Thrun et al. 2005, Chapter 5) and a proportional control law for following a planned trajectory. Laser beams are simulated through ray-casting operation over the ground truth 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.

Table 3: Parameters for frontier and MI maps computations. Note that the employed maximum sensor range and the maximum range used in the MI algorithm for prediction do not need to be the same.

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 and size of TestDataWindow	r_{max}	
– Intel map		14.0 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
Max range	r_{max}	
– Intel map		4.0 m
– Freiburg map		60.0 m
Numerical integration resolution	s_z	
– Intel map		$10/3 \text{ m}^{-1}$
– Freiburg map		1 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

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 ground truth 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 3. The sensitivity of the parameters in Table 3 is not high and slight variations of them ($\sim 10\%$) 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). As described in Section 4.3, 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

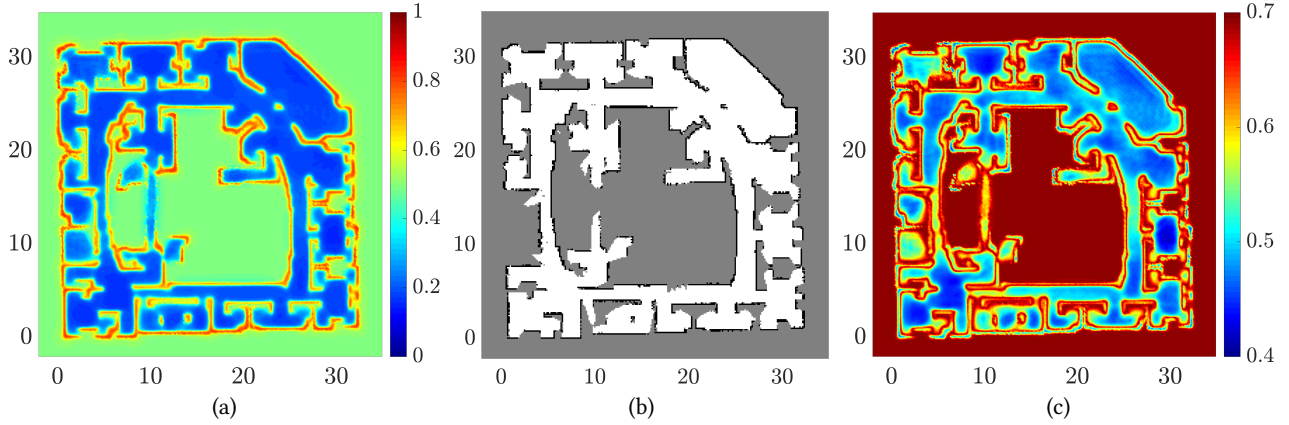


Figure 8: MI-based exploration in the Intel map derived from the Intel dataset. (a) I-GPOM2, (b) the equivalent OGM computed at the end of the experiment (c) corresponding entropy map of the GPOM (nats). The sparse observations due to the occluded perception field in a complex environment such as the Intel map signifies the capabilities of OGM and GPOM methods to cope with such limitations. Map dimensions are in meters, and the maps are built with the resolution 0.135 m.

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). The map entropy at any time-step can be computed using (21). The map entropy calculation can become independent of the map resolution following the idea in [Stachniss et al. \(2005\)](#); that is the cell area, i.e. the squared of the map resolution, weights each entropy term. To see the performance of decision-making across the entire an experiment, the MER is then computed at the end of each experiment using the difference between final and initial map entropies divided by the number of exploration steps. 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.

5.2 Exploration results in the Intel map

An example of the exploration results using GPOMI is shown in Figures 8 and 9. The statistical summary of the results are depicted in Figure 10. 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.

GPOMF and GPOMI 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. The results from

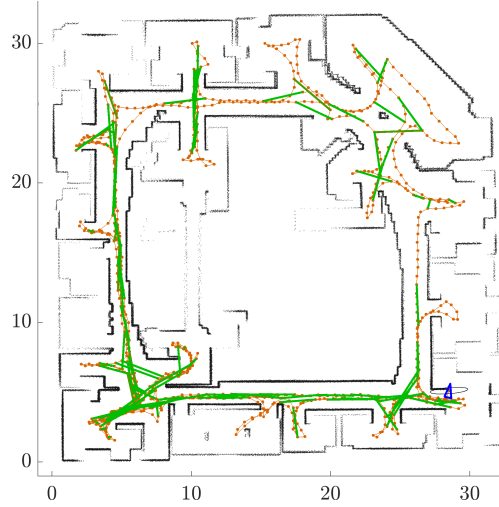


Figure 9: Pose SLAM map of the MI-based exploration in the Intel map derived from the Intel dataset. Dotted (red) curves are the robot path and connecting lines (green) indicate loop-closures. Map dimensions are in meters. The starting robot position is at (18,26), horizontally and vertically, respectively, and the robot terminates the exploration mission at the most bottom right room.

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 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 Outdoor scenario: Freiburg Campus

In the second scenario, the map is an outdoor area with a larger size (almost ten times). Figure 11 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 results is shown in Figure 12. To maintain the computational time manageable, the occupancy maps are built with the coarse resolution of 1 m.

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. 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

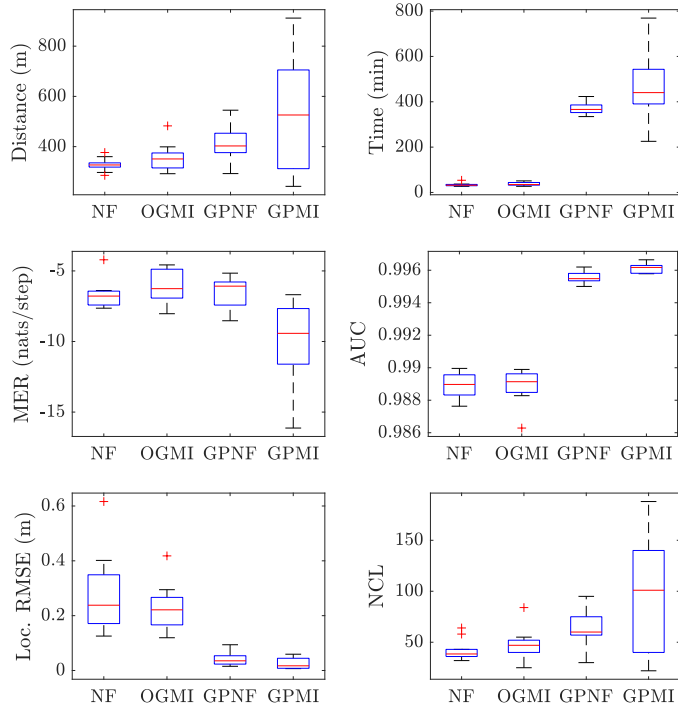


Figure 10: The box plots show comparison of different exploration strategies in the Intel dataset from 10 independent runs. The compared criteria are travel distance (m), time (min), map entropy rate (nats/step), the mapping performance using the area under the receiving operating characteristic curve,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



Figure 11: 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.

idea of implicit loop-closing actions due to the correlations between the map and the robot pose. However, the NCL distribution has wider tails which does not support its repeatability. The exploration times in this environment is less than those of the previous experiment in the Intel map. We associate the faster

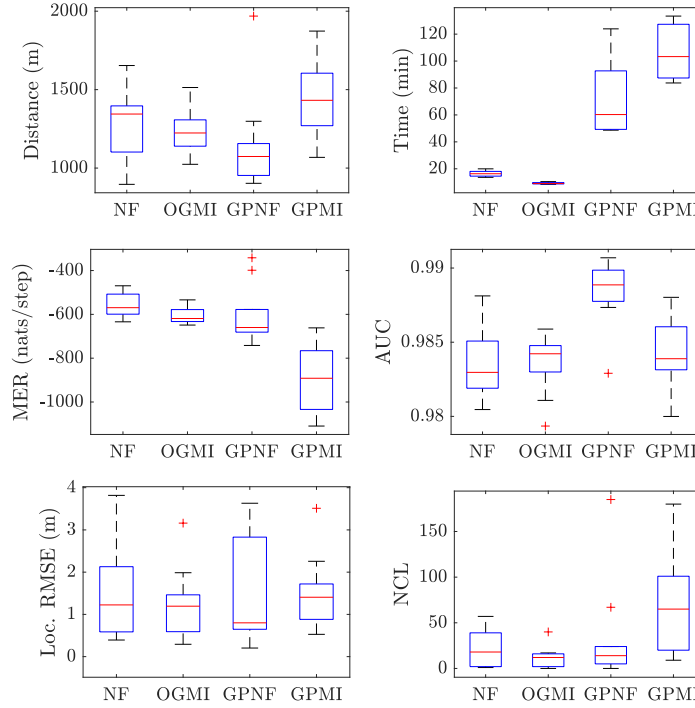


Figure 12: The box plots show comparison of different exploration strategies in the Freiburg campus dataset from 10 independent runs. The compared criteria are travel distance (m), time (min), map entropy rate (nats/step), the mapping performance using the area under the receiving operating characteristic curve, localization root mean-squared error (m), and the number of closed loops by Pose SLAM.

map exploration results with the combination of the difference in map resolutions and the open shape of the Freiburg campus map. In contrast, the Intel map is highly structured with narrow hallways and small rooms which require a finer map resolution leading to a higher number of query points. Furthermore, in the Intel map, unlike the Freiburg campus map, a larger maximum range does not help the robot to explore the map faster due to the occlusion problem.

Figure 13 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 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.

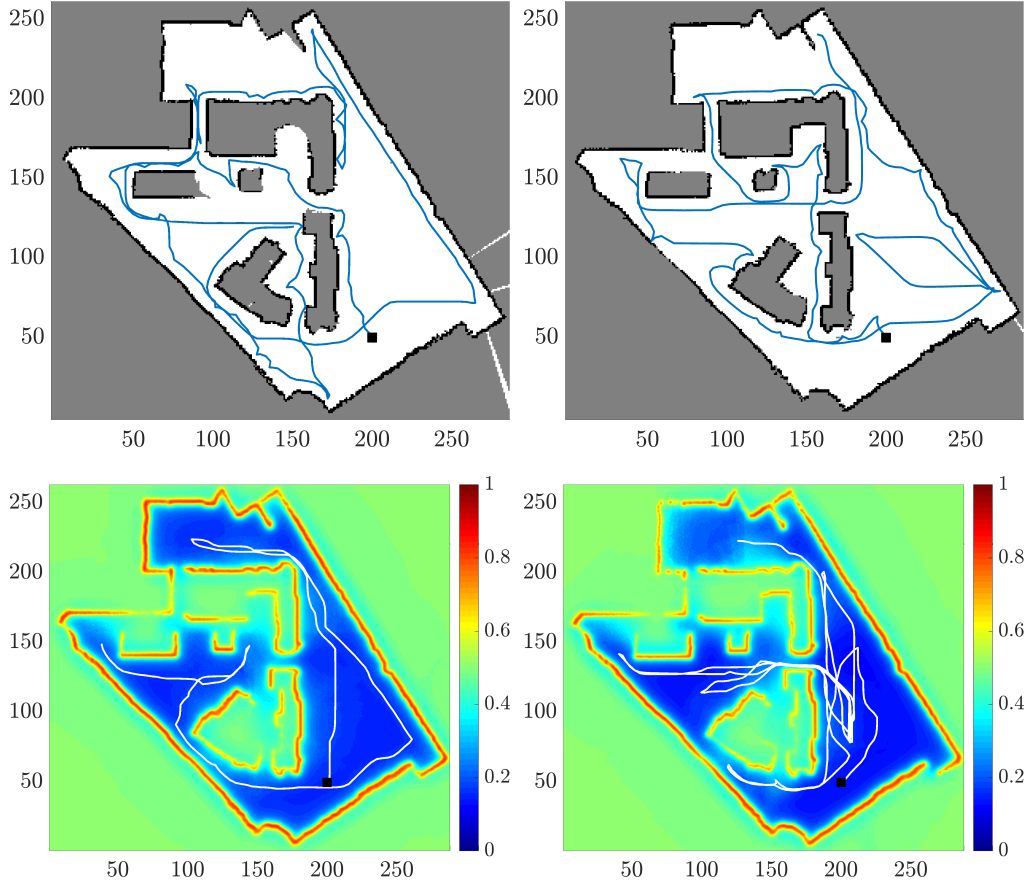


Figure 13: Illustrative examples of exploration in the Freiburg Campus map. The top left and right, and the bottom left and right figures show the results for NF, OGMI, GPNF, and GPMI, respectively.

6 Conclusion and Future Work

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.

Extensions of this work include development of the planning algorithms with longer horizons as well as incorporating the robot pose uncertainty into the mapping and decision-making frameworks (Ghafari Jadidi et al. 2016, 2017b). Furthermore, as analyzed and discussed in Subsection 3.7, development of computationally more attractive GPOM algorithms remains as an interesting future direction.

References

- Amigoni F and Caglioti V (2010) An information-based exploration strategy for environment mapping with mobile robots. *Robot. Auton. Syst.* 58(5): 684–699. 6
- Bajcsy R (1988) Active perception. *Proceedings of the IEEE* 76(8): 966–1005. 6
- Binney J and Sukhatme GS (2012) Branch and bound for informative path planning. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 2147–2154. 3
- Blanco JL, Fernandez-Madrigal JA and González J (2008) A novel measure of uncertainty for mobile robot SLAM with rao-Blackwellized particle filters. *The Int. J. Robot. Res.* 27(1): 73–89. 6
- 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. 6
- 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. 6
- 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. 6
- Carlone L, Du J, Ng MK, Bona B and Indri M (2014) Active slam and exploration with particle filters using kullback-leibler divergence. *Journal of Intelligent & Robotic Systems* 75(2): 291–311. 6
- Carrillo H, Dames P, Kumar V and Castellanos JA (2015) Autonomous robotic exploration using occupancy grid maps and graph SLAM based on Shannon and Rényi entropy. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 487–494. 6
- Carrillo H, Reid I and Castellanos JA (2012) On the comparison of uncertainty criteria for active SLAM. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 2080–2087. 6
- Charrow B (2015) *Information-theoretic active perception for multi-robot teams*. PhD Thesis, University of Pennsylvania. 7
- 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. 7

- 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. 7, 25
- Cover TM and Thomas JA (1991) *Elements of information theory*. John Wiley & Sons. 18
- 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. 6
- Elfes A (1987) Sonar-based real-world mapping and navigation. *Robotics and Automation, IEEE Journal of* 3(3): 249–265. 3, 6
- Faigl J, Kulich M and Preučil L (2012) Goal assignment using distance cost in multi-robot exploration. In: *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* IEEE, pp. 3741–3746. 7
- Fawcett T (2006) An introduction to roc analysis. *Pattern recognition letters* 27(8): 861–874. 12
- Feder HJS, Leonard JJ and Smith CM (1999) Adaptive mobile robot navigation and mapping. *The Int. J. Robot. Res.* 18(7): 650–668. 6
- Ghaffari Jadidi M, Gan L, Parkison SA, Li J and Eustice RM (2017a) Gaussian processes semantic map representation. In: *RSS Workshop on Spatial-Semantic Representations in Robotics*. 3
- 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. 3, 7, 8, 9, 22
- 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. 3, 4
- Ghaffari Jadidi M, Valls Miro J and Dissanayake G (2016) Sampling-based incremental information gathering with applications to robotic exploration and environmental monitoring. *arXiv* 1607.01883. URL <http://arxiv.org/abs/1607.01883>. 3, 29
- Ghaffari Jadidi M, Valls Miro J and Dissanayake G (2017b) Warped Gaussian processes occupancy mapping with uncertain inputs. *IEEE Robotics and Automation Letters* 2(2): 680–687. 3, 29
- 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*. 3, 7, 8
- 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.* 3, 7
- González-Banos H and Latombe J (2002) Navigation strategies for exploring indoor environments. *The Int. J. Robot. Res.* 21(10-11): 829–848. 21
- 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. 7

- He R, Brunskill E and Roy N (2010) Puma: Planning under uncertainty with macro-actions. In: *AAAI*. 4
- Hensman J, Fusi N and Lawrence ND (2013) Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*. 17
- Hollinger GA (2015) Long-horizon robotic search and classification using sampling-based motion planning. In: *Robotics: Science and Systems*. 3
- Hollinger GA and Sukhatme GS (2014) Sampling-based robotic information gathering algorithms. *The Int. J. Robot. Res.* 33(9): 1271–1287. 3
- 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. 6
- Howard A and Roy N (2003) The robotics data set repository (Radish). URL <http://radish.sourceforge.net>. 12, 16, 19, 21, 26
- Ila V, Porta J and Andrade-Cetto J (2010) Information-based compact Pose SLAM. *IEEE Trans. Robot.* 26(1): 78–93. 3, 6, 22
- 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. 6
- Keidar M and Kaminka GA (2013) Efficient frontier detection for robot exploration. *The Int. J. Robot. Res.* DOI: 10.1177/0278364913494911. 17
- 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. 6
- 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. 3, 7
- 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. 3, 13
- 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. 3, 8
- 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. 3, 8
- Konolige K (1997) Improved occupancy grids for map building. *Auton. Robot* 4(4): 351–367. 6
- Krause A and Guestrin C (2005) Near-optimal value of information in graphical models. In: *Conference on Uncertainty in Artificial Intelligence (UAI)*. 3
- Krause A and Guestrin C (2007) Nonmyopic active learning of gaussian processes: an exploration-exploitation approach. In: *Proceedings of the 24th international conference on Machine learning*. ACM, pp. 449–456. 3
- Lang T, Plagemann C and Burgard W (2007) Adaptive non-stationary kernel regression for terrain modeling. In: *Robotics: Science and Systems*. 7

- Lin J (1991) Divergence measures based on the Shannon entropy. *IEEE Trans. Information Theory* 37(1): 145–151. 21
- 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. 6
- Merali RS and Barfoot TD (2014) Optimizing online occupancy grid mapping to capture the residual uncertainty. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 6070–6076. 6
- Moravec HP and Elfes A (1985) High resolution maps from wide angle sonar. In: *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2. IEEE, pp. 116–121. 3, 6
- Murphy KP (2012) *Machine learning: a probabilistic perspective*. The MIT Press. 9
- 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. 14
- 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. 3, 7
- Pukelsheim F (2006) *Optimal design of experiments*. Society for Industrial and Applied Mathematics. 6
- Ramos F and Ott L (2015) Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent. In: *Robotics: Science and Systems*. 7
- Rasmussen C and Williams C (2006) *Gaussian processes for machine learning*, volume 1. MIT press. 7, 8, 9, 23
- Russell SJ and Norvig P (2009) *Artificial intelligence: a modern approach*. 3 edition. Prentice Hall. 3
- Sim R and Roy N (2005) Global A-optimal robot exploration in SLAM. In: *Proc. IEEE Int. Conf. Robot Automat.* IEEE, pp. 661–666. 6
- Singh A, Krause A, Guestrin C and Kaiser WJ (2009) Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research* : 707–755. 3
- Snelson E and Ghahramani Z (2006) Sparse Gaussian processes using pseudo-inputs. In: *Advances in Neural Information Processing Systems* 18. pp. 1257–1264. 17
- Stachniss C, Grisetti G and Burgard W (2005) Information gain-based exploration using Rao-Blackwellized particle filters. In: *Robotics: Science and Systems*, volume 2. 6, 24
- Stachniss C, Plagemann C, Lilienthal AJ and Burgard W (2008) Gas distribution modeling using sparse gaussian process mixture models. In: *Robotics: Science and Systems*. 3
- Stein M (1999) *Interpolation of spatial data: some theory for kriging*. Springer Verlag. 8
- 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. 21
- T O’Callaghan S and Ramos F (2012) Gaussian process occupancy maps. *The Int. J. Robot. Res.* 31(1): 42–62. 3, 7, 13

- Thrun S (2003) Learning occupancy grid maps with forward sensor models. *Autonomous robots* 15(2): 111–127. 6
- Thrun S, Burgard W and Fox D (2005) *Probabilistic robotics*, volume 1. MIT press. 4, 22, 23
- Tresp V (2000) A Bayesian committee machine. *Neural Computation* 12(11): 2719–2741. 7, 12
- Valencia R, Morta M, Andrade-Cetto J and Porta JM (2013) Planning reliable paths with Pose SLAM. *IEEE Trans. Robot.* 29(4): 1050–1059. 6
- 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. 6
- Vallvé J and Andrade-Cetto J (2013) Mobile robot exploration with potential information fields. In: *6th European Conference on Mobile Robots*. 6
- 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. 6
- Vallvé J and Andrade-Cetto J (2015) Potential information fields for mobile robot exploration. *Robotics and Autonomous Systems* 69: 68–79. 6
- 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. 3, 7
- Wang J and Englot B (2016) Fast, accurate Gaussian process occupancy maps via test-data octrees and nested Bayesian fusion. In: *Proc. IEEE Int. Conf. Robot Automat.* pp. 1003–1010. 7
- Yamauchi B (1997) A frontier-based approach for autonomous exploration. In: *Int. Sym. Comput. Intell. Robot. Automat.* pp. 146–151. 3, 6, 21
- 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. 7