Gaussian Process Autonomous Mapping and Exploration for Range Sensing Mobile Robots

Maani Ghaffari Jadidi† Jaime Valls Miro† Gamini Dissanayake†

May 3, 2016

Abstract

A framework for incremental autonomous mapping and exploration of unknown environments using the recently emerged Gaussian Process (GP) continuous occupancy mapping techniques is proposed in this article. The technique is able to exploit structural dependencies present in the environment as well as handle sparse sensor measurements. A strategy based on mutual information surfaces between the current map of the environment and predicted sensor readings using a one-step look ahead and macro-action concept is used to generate the control actions. We present results using the publicly available Intel Research Lab dataset with maps generated with occupancy grid-based nearest frontier, GP-based nearest frontier, and the proposed GP and mutual information-based exploration techniques. Maps are compared using the receiver operating characteristic curve and the area under the curve to demonstrate the accuracy of the proposed incremental mapping technique. Statistical significance test using two-sample t-test demonstrates the effectiveness of the proposed technique in terms of map entropy reduction rate and produced map quality (p ≤ 0.05).

∗Working paper. mani.ghaffari@gmail.com – http://maanighaffari.com
†Centre for Autonomous Systems (CAS), University of Technology Sydney.
## Contents

1 Introduction ............................................... 3

2 Related work ............................................. 5

3 Mapping ................................................... 6

   3.1 Gaussian Processes ................................. 6
   3.2 Problem statement and formulation .............. 7
   3.3 Sensor model, training and test data .......... 8
   3.4 Map management .................................. 9
   3.5 I-GPOM2; an improved mapping strategy ..... 11
   3.6 Frontier map ....................................... 12

4 Exploration .............................................. 15

   4.1 Mutual information computation ............... 15
   4.2 Decision making .................................. 17

5 Results and Discussion ................................. 17

   5.1 Experiments setup ................................. 18
   5.2 Exploration results in the Intel map ......... 19
   5.3 Statistical analysis of the results ............ 20
   5.4 Computational complexity ....................... 21

6 Conclusion .............................................. 22
1 Introduction

Autonomous mobile robots are required to generate a spatial representation of the robot environment, known as the mapping problem. This 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 to 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. In this article we are concerned with autonomous exploration when the location of the robot is available through an appropriate strategy such as Pose SLAM [17].

Information gain-based methods that minimize entropy-based cost functions have been proposed for solving the autonomous exploration problem [8, 3, 38, 4, 1]. Commonly employed grid-based occupancy mapping techniques [28, 6] to represent the environment rely on the assumption of independence between cells to infer the map posterior through marginalization and as such ignore the structural dependencies in the environment. In contrast, Gaussian Process (GP) based occupancy maps [31, 22, 11] 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 [18] to improve the overall performance of the exploration algorithms. Mutual information (MI) is guaranteed to be submodular and non-decreasing on any subset of the union of the state space and observation space given conditional independence assumption between observations [25]. Moreover, there is a proven lower-bound for sub-optimality of the greedy algorithm for maximizing such functions [29]. Therefore, it is reasonable to choose such a function for the utility optimization problem. 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 for planning under uncertainty [14] and define it as an exploration target (frontier) which is assumed to be reachable through an open-loop control strategy.

In this article, a novel method for autonomous exploration based on computation of MI surfaces that builds on the GP-based mapping and exploration technique proposed in [11] is presented. The developed exploration method uses a probabilistic frontier representation and continuous occupancy maps and is able to handle sparse observations. Furthermore, the MI values are computed accurately using a forward sensor model map prediction algorithm. Figure 1 depicts the proposed navigation process concept using GPs maps. Given the robot pose and corresponding laser scans, Gaussian Process occupancy map (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 [41], however, the proposed algorithm can be adapted to other sensor modalities with reasonable probabilistic observation models.

Contributions

This article is based on our preliminary work [12] 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:
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 navigations tasks as well as cost-aware planning. The explorer node returns a macro-action (chosen frontier) that optimizes the expected utility function.

− We expand the decision making part by the notion of macro-action, and calculate the travel cost as the actual traversable path length to each frontier.

− 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, including a statistical significance test with comparable techniques using two-sample t-test.

The followings are the main contributions of this work:

− We present a rigorous formulation for the GP occupancy mapping and a reproducible algorithmic implementation of the proposed exploration methodology.

− We demonstrate that the accuracy of the incremental mapping method suggested in this work is comparable to that of batch GP occupancy mapping, while consuming less computational time.

− We provide in-depth analysis of results gathered using the publicly available Intel Research Lab dataset [16], together with a statistical significance test with respect to comparable exploration techniques in the literature.
Notation

In this article probabilities and probability densities are not distinguished in general. Matrices are capitalized in bold and vectors are in bold type. With a slight abuse of notation, for the sake of compactness, random variables and their realizations are denoted interchangeably and can represent a set of random variables where it is evident from context.

Outline

The remaining parts of this article are organized as follows. In Section 2, a literature review is given. In Section 3, we present and compare the proposed mapping algorithms. 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 with the Intel dataset. 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 to “frontiers” that indicate unknown regions of the environment in the neighborhood of known free areas [46]. A combined information utility for exploration was developed in [3] using the information-based cost function in [8] and an occupancy grid map (OGM). A one step look-ahead strategy was 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 [27] an integrated exploration approach for a robot navigating in an unknown environment populated with beacons was proposed. This method examined the exploration problem through a total utility function consisting of the weighted sum of the OGM entropy, navigation cost, and localizability. The vehicle pose was assumed to follow a Gaussian distribution, thus the entropy was proportional to the determinant of the vehicle pose covariance. In order to enhance the map quality of the extended Kalman filter-based simultaneous localization and mapping (SLAM), an a-optimal criterion for autonomous exploration in [36] was examined. To simplify the objective function, the map covariance matrix was approximated by ignoring the correlation among the features. Furthermore, the search space of the trajectories was limited by employing a breadth-first search algorithm to find global trajectories leading to a tractable method.

In [38], Rao-Blackwellized particle filters were employed to compute map and robot pose posteriors. The proposed uncertainty reduction approach was based on the joint entropy minimization of the SLAM posterior. The information gain was approximated using ray-casting for a given action. In a similar framework, 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 was addressed in [4]. [1] assumed that all random variables are normally distributed and proposed an exploration strategy based on relative entropy metric, combined traveling cost and expected information gain.

The techniques in [43, 45] 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 [17, 44], a delayed-state SLAM algorithm.
from the pose-graph family. Given the inherent complexity in the formulation to calculate joint entropy of robot pose and map, conditional independence was assumed.

The aforementioned methods fall short of accounting for structural correlations in the environment. Non-parametric kernel models, such as GPs, have proven particularly powerful to represent the affinity of spatially correlated data, overcoming the assumption of independence between cells, together with the possibility of multi-resolution queries [32, 31]. 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 search strategies for the robot. 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 [47]. In line with the idea of high-dimensional map building, in [33], Hilbert maps technique suggests to use a logistic regression classifier to operate on a high-dimensional feature vector that projects measurements into a reproducing kernel Hilbert space. The method is more scalable and can be updated in linear time, however, it is an approximation for continuous occupancy mapping (COM) and it needs to be adopted for navigation tasks.

In [19], the MI surface between a map and future measurements was 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 was formally proven that any controller tasked to maximize an MI reward function is eventually attracted to unexplored areas.

In the present work, a tractable method for autonomous exploration based on computation of the MI surface is presented. The novel 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. Moreover, the work is thoroughly demonstrated with examples under realistic conditions.

3 Mapping

GPOM, in its original formulation [32, 31], 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 incremental GP map building was studied in [21], and in [10, 9, 11]. 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 Bayesian Committee Machine (BCM) technique [42].

3.1 Gaussian Processes

A Gaussian Process is a collection of any finite number of random variables which are jointly distributed Gaussian [34]. We define a training set \( D = \{ (\mathbf{x}^{[i]}, y^{[i]}) | i = 1, ..., n \} \) in which consists of a \( d \)-dimensional input vector \( \mathbf{x} \) and a scalar output (target) value \( y \) for \( n \) observations. The joint distribution of the observed target values, \( y = [y^{[1]}, ..., y^{[n]}]^T \), and the function values (the latent variable), \( f^* \), at the query points can be written as

\[
\begin{bmatrix}
y \\
f^*
\end{bmatrix}
= \mathcal{N}(0, \begin{bmatrix} K(\mathbf{X}, \mathbf{X}) + \sigma_n I_n \times n & K(\mathbf{X}, \mathbf{X}^*) \\ K(\mathbf{X}^*, \mathbf{X}) & K(\mathbf{X}^*, \mathbf{X}^*) \end{bmatrix})
\] (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, \( \mathbf{I}_{n \times n} \) is the \( n \)-by-\( n \) identity matrix, and \( \sigma_n \) 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 \( \mathbf{x}_* \) is given by

\[
\mu = \bar{f}_* = k(\mathbf{X}, \mathbf{x}_*)^T \mathbf{K}(\mathbf{X}, \mathbf{X})^{-1} \mathbf{y} \\
\sigma = \text{var}[f_*] = k(\mathbf{x}, \mathbf{x}_*) - k(\mathbf{X}, \mathbf{x}_*)^T \mathbf{K}(\mathbf{X}, \mathbf{X})^{-1} k(\mathbf{X}, \mathbf{x}_*)
\]

The Matérn family of covariance functions [39] has proven powerful features to model structural correlations [10, 23, 11, 24] and hereby we select them as the kernel of GPs. For a single query point the function is given by

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

where \( \Gamma \) is the Gamma function, \( K_{\nu}(\cdot) \) is the modified Bessel function of the second kind of order \( \nu \), \( \kappa \) is the characteristic length scale, and \( \nu \) is a parameter used to control the smoothness of the covariance.

### 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 \( \mathcal{M} = (M^1, \ldots, M^{n_m}) \) whose elements are describe by a normal distribution \( m^{[i]} \sim \mathcal{N}(\mu^{[i]}, \sigma^{[i]}) \), \( i \in \{1, \ldots, n_m\} \). 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 \( (Z^1, \ldots, Z^{n_z}) \) whose elements can take values \( z^{[k]} \in \mathcal{Z} \), \( k \in \{1, \ldots, n_z\} \). Let \( \mathcal{X} \subset \mathbb{R}^2 \) be the set of spatial coordinates we are interested to build a map on. Let \( \mathbf{x}_y^{[k]} \in \mathcal{X}_o \subset \mathcal{X} \) be an observed occupied point by \( k \)-th sensor beam from the environment which can be calculated by transforming the local observation \( z^{[k]} \) to the global frame using the robot pose. 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 \( \mathcal{D}_o = \{ (\mathbf{x}_o^{[i]}, y_o^{[i]} ) | i = 1, \ldots, n_o \} \) and a training set of unoccupied points \( \mathcal{D}_f = \{ (\mathbf{x}_f^{[i]}, y_f^{[i]} ) | i = 1, \ldots, n_f \} \) in which \( y_o = [y_o^{[1]}, \ldots, y_o^{[n_o]}]^T \) and \( y_f = [y_f^{[1]}, \ldots, y_f^{[n_f]}]^T \) are target vectors and each of their elements can belong to the set \( \mathcal{Y} = \{-1, +1\} \subset \mathcal{M} \) where \( -1 \) and \( +1 \) corresponds to unoccupied and occupied locations, respectively, \( n_o \) is total number of occupied points, and \( n_f \) is total number of unoccupied points. Given the
robot pose $x_t \in \text{SE}(2)$ and observations $Z_t = z_t$, we are interested to estimate $p(M = m \mid x_t, Z_t = z_t)$. The map can be inferred as a Gaussian Process by defining the process as the function $y(x) : \mathcal{X} \to \mathcal{M}$, therefore

$$y(x) \sim \mathcal{GP}(m(x), k(x, x'))$$

(5)

It is often the case that we set the mean function $m(x)$ as zero, unless it is mentioned explicitly that $m(x) \neq 0$. For a given query point in the map, $x_s$, GP predicts a mean, $\mu_s$, and an associated variance, $\sigma_s$. We can write

$$m[i] = y(x_s[i]) \sim \mathcal{N}(\mu_s[i], \sigma_s[i])$$

(6)

In order to show a valid probabilistic representation of the map $p(m[i])$, the classification step 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 obstacles 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. In order 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 $z_t = (z_t[1], ..., z_t[n_z])$ has $n_z$ range observations at a specific bearing depending on the density of the beam. The observation model for each $z_t[k]$ can be written as

$$z_t[k] = \begin{bmatrix} r_t[k] \\ \alpha_t[k] \end{bmatrix} = h(x_t, x_o[k]) + v, \quad v \sim \mathcal{N}(0, R)$$

(7)
\[ h(x_t, x_o^{[k]}) \triangleq \left[ \arctan(x_o^{[k,2]} - x_o^{[k,1]}, x_o^{[k,1]} - x_o^{[k,3]}) \right]
\left[ \sqrt{(x_o^{[k,1]} - x_o^{[k,1]})^2 + (x_o^{[k,2]} - x_o^{[k,2]})^2} \right] \]

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 \( \mathbf{R} \). We are interested to know the value of \( x_o^{[k]} \), which is in the map space, however, the sensor measures \( r_t^{[k]} \) and \( \alpha_t^{[k]} \).

The inverse model can be calculated as

\[ x_o^{[k]} = x_t^{[1:2]} + r_t^{[k]} R(x_t^{[3]}) \begin{bmatrix} \cos(\alpha_t^{[k]}) \\ \sin(\alpha_t^{[k]}) \end{bmatrix} \]

where \( R(x_t^{[3]}) \) indicates a 2 x 2 rotation matrix.

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

\[ x_f^{[k,j]} = x_t^{[1:2]} + \delta_j \begin{bmatrix} \cos(\alpha_t^{[k]}) \\ \sin(\alpha_t^{[k]}) \end{bmatrix} \]

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 \( r_t^{[k]} = \|l_e^{[k]}\| \). In 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

\[ n_f^{[k]} = \max(n_{f,min}, s_t(r_t^{[k]})) \]

where \( s_t(.) \) is a function that adaptively generates a number 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 \( \mathcal{D}_f^{[i]} = \{ (x_f^{[k]}, y_f^{[k]}), \quad k = 1, ..., n_z \} \) and \( y_f^{[k]} = [y_f^{[1]}, ..., y_f^{[n_f^{[k]}]}]^T \).

Test points can have any desired spatial distribution. However building the map over a grid, as implemented in the results section, facilitates comparison with standard occupancy grid-based methods – i.e., at similar map resolutions.

### 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.

In order to incorporate new information incrementally, map updates are performed using BCM. The
Figure 4: Comparison of the incremental (I-GPOM) and batch (GPOM) COM 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.

technique combines estimators which were trained on different data sets. Assuming a Gaussian prior with zero mean and covariance \( \Sigma \) and each GP with mean \( E[f_{\text{s}}|D[i]] \) and covariance \( \text{cov}[f_{\text{s}}|D[i]] \), it follows that \[42\]
\[
E[f_{\text{s}}|D] = C^{-1} \sum_{i=1}^{p_m} \text{cov}[f_{\text{s}}|D[i]]^{-1} E[f_{\text{s}}|D[i]]
\]
\[
C = \text{cov}[f_{\text{s}}|D]^{-1} = -(p_m - 1)(\Sigma)^{-1} + \sum_{i=1}^{p_m} \text{cov}[f_{\text{s}}|D[i]]^{-1}
\]
where \( p_m \) is the total number of mapping processes.

In Figure 4 a comparison of the incremental (I-GPOM) and batch (GPOM) GP occupancy mapping using the Intel dataset 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 \[7\]. Without loss of generality, an observation size of 25 laser scans had to be set due to the memory limitation imposed by
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        |

Table 2: Conceptual comparison of different occupancy mapping methods for mapping and exploration applications. The + and – signs correspond to the advantage or disadvantage of an algorithm for that item.

|                      | OGM | I-GPOM | I-GPOM2 |
|----------------------|-----|--------|---------|
| Time complexity      | +   | –      | –       |
| Resolution           | –   | +      | +       |
| Sparse observations  | –   | +      | +       |
| Mapping              | +   | +      | +       |
| Exploration          | +   | –      | +       |

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.

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 [31] and Figure 3 in [22]). 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 [31]).

The first problem is directly related to the inferred map quality, whilst the second is a challenge for exploration using continuous occupancy maps. The integral kernel approach [30] 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
Figure 5: Occupancy maps visualization; from left to right: OGM, I-GPOM, I-GPOM2. The maps are build incrementally using all the observations available in the Intel dataset. The regressed maps, I-GPOM and I-GPOM2, are both able to complete partially observable areas – i.e., incomplete areas in the OGM, however, using two GPs in I-GPOM2 method produces more accurate maps. The SLAM problem is solved by using Pose SLAM, thus map quality depends on SLAM results.

Algorithm 1 IGPOM()

Require: Robot pose $p$, measurements $z$
1: if firstFrame then
2: \[ m = \emptyset \]
3: optimize GP hyperparameters $\theta$
4: end if
5: $X_{*} \leftarrow$ TestDataWindow($p$)
6: $X_{o}, y_{o} \leftarrow$ Transform2Global($p, z$)
7: $X_{f}, y_{f} \leftarrow$ TrainingData($p, z$)
8: $[\mu_{*}, \sigma_{*}] \leftarrow$ GP($\theta, [X_{o}; X_{f}], [y_{o}; y_{f}], X_{*}$)
9: $m \leftarrow$ UpdateMap($\mu_{*}, \sigma_{*}, m$)
10: return $m$

more flexibility to model the cluttered rooms and outperforms the other methods in map accuracy. The ground truth map was generated using the registered points map and an image dilation technique to remove outliers. Table 2 summarizes the advantages and disadvantages of the three discussed occupancy mapping methods and their applicability for mapping and exploration scenarios.

The steps of the incremental GPOM (I-GPOM) are shown in Figure 2 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.

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 classical binary representation, defining frontiers in probabilistic terms as distributions over informative areas using map uncertainty is more suitable for computing expected behaviors.
Algorithm 2 IGPO\textsubscript{2}()

\textbf{Require:} Robot pose $p$, measurements $z$:

1: \textbf{if} firstFrame \textbf{then}
2: \hspace{1em} $m = m_o = m_f = \emptyset$
3: \hspace{1em} optimize GP hyperparameters $\theta_o, \theta_f$
4: \textbf{end if}
5: \hspace{1em} $X_\ast \leftarrow \text{TestDataWindow}(p)$
6: \hspace{1em} $X_o, y_o \leftarrow \text{Transform2Global}(p, z)$
7: \hspace{1em} $X_f, y_f \leftarrow \text{TrainingData}(p, z)$
8: \hspace{1em} $[\mu_o, \sigma_o] \leftarrow \text{GP}(\theta_o, X_o, y_o, X_\ast)$
9: \hspace{1em} $[\mu_f, \sigma_f] \leftarrow \text{GP}(\theta_f, X_f, y_f, X_\ast)$
10: $m_o \leftarrow \text{UpdateMap}(\mu_o, \sigma_o, m_o)$
11: $m_f \leftarrow \text{UpdateMap}(\mu_f, \sigma_f, m_f)$
12: $m \leftarrow \text{MergeMap}(m_o, m_f)$
13: \textbf{return} $m, m_o$

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

1: $\sigma_c \leftarrow (\sigma_a^{-1} + \sigma_b^{-1})^{-1}$
2: $\mu_c \leftarrow \sigma_c (\sigma_a^{-1} \mu_a + \sigma_b^{-1} \mu_b)$
3: \textbf{return} $\mu_c, \sigma_c$

Algorithm 4 UpdateMap()

\textbf{Require:} Global map $m$, local map $m_\ast$:

1: \textbf{for} all $i \in M_\ast$ \textbf{do}
2: \hspace{1em} $j \leftarrow \text{find the global map index for } i \text{ using a nearest neighbor search}$
3: \hspace{1em} $\mu[i], \sigma[i] \leftarrow \text{FusionBCM}(\mu[j], \mu[i], \sigma[j], \sigma[i])$
4: \textbf{end for}
5: $m \leftarrow \text{LogisticRegression}(\mu, \sigma)$
6: \textbf{return} $m$

Algorithm 5 MergeMap()

\textbf{Require:} Current unoccupied $m_f$ and occupied $m_o$ maps estimates:

1: \textbf{for} all $i \in M$ \textbf{do}
2: \hspace{1em} $\mu[i], \sigma[i] \leftarrow \text{FusionBCM}(\mu_o[i], \mu_f[i], \sigma_o[i], \sigma_f[i])$
3: \textbf{end for}
4: $m \leftarrow \text{LogisticRegression}(\mu, \sigma)$
5: \textbf{return} $m$

The mean value of a frontier point can be computed as

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

(14)

where $\nabla$ denotes the gradient operator, and $\beta$ 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 map
Figure 6: Inferred continuous occupancy map (left); associated probabilistic frontier map (middle); and mutual information surface (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. 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.

Algorithm 6 \textbf{BuildFrontierMap()}

\textbf{Require:} Current map $m$ and occupied map $m_o$ estimates;
\begin{enumerate}
    \item $dm \leftarrow \|\nabla p(m)\|_1$, $dm_o \leftarrow \|\nabla p(m_o)\|_1$
    \item $\sigma_{\text{min}} \leftarrow \min(\sigma)$
    \item $f \leftarrow \emptyset$
    \item \textbf{for} all $i \in \mathcal{M}$ \textbf{do}
    \begin{enumerate}
        \item $f[i] \leftarrow dm[i] - \beta(dm_o[i] + m_o[i] - 0.5)$
        \item $w[i] \leftarrow \gamma \sqrt{\lambda[i]/\sigma[i]}$
        \item $f[i] \leftarrow (1 + \exp(-w[i]f[i]))^{-1}$
    \end{enumerate}
    \item \textbf{end for}
\end{enumerate}
\textbf{return} $f$

uncertainty. In order to squash mean and variance values into the range $[0, 1]$, a logistic regression classifier with inputs from $f[i]$ and map uncertainty $\sigma[i]$ is applied to data which yields

$$p(f[i]|m[i], w[i]) = \frac{1}{1 + \exp(-w[i]f[i])}$$

(15)

where $w[i] = \gamma \sqrt{\lambda[i]}$ denotes the required weights, $\lambda[i] = \sigma_{\text{min}}/\sigma[i]$ is the bounded information associated to location $i$, $\gamma > 0$ is a constant to control the sigmoid shape, and $\sigma_{\text{min}}$ is the minimum estimated variance by the GP. 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. The aforementioned representation suggests a compatible interface with other modules of our Bayesian approach and provides a meaningful way to set an automatic stop criterion for exploration experiments.
4 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 [27, 38, 35, 43, 19].

In its general form, this process can be formulated as a partially observable Markov decision process [20] or an optimal control problem with imperfect state information [2]. In the case of robot navigation, the state and action space are often continuous. As such, it is computationally expensive to address substantial problems. Fortunately, due to the submodularity and non-decreasing properties of the MI for the problem defined in this article, it is possible to consider the next-best view problem and yet achieve a near-optimal solution that in the worst-case can attain about 63% of the utility function optimal value [25, 29]. 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 estimate the map posterior with a forward sensor model through the Bayes update formula and compute MI numerically. The COM technique infers a joint predictive distribution over the map, therefore, the computed map entropy is more descriptive of the real map uncertainty (see Figure 1 in [19] for an illustration of map entropy vs. MI in OGMs).

4.1 Mutual information computation

MI is the reduction in uncertainty of a random variable due to the knowledge of another random variable [5]. In other words, assuming robot poses are known, given a measurement \( Z = z \) from \( Z \) what will be the reduction in the map \( M = m \) uncertainty? The MI between the map and the future measurement \( Z_{t+1} = \hat{z} \) is

\[
I(M; Z_{t+1}|z_{1:t}) = \int_{\hat{z} \in Z} \sum_{m \in M} p(m, \hat{z}|z_{1:t}) \log \frac{p(m, \hat{z}|z_{1:t})}{p(m|z_{1:t})p(\hat{z}|z_{1:t})} d\hat{z}
\]

\[
= H(M|z_{1:t}) - \overline{H}(M|Z_{t+1}, z_{1:t}),
\]

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

\[
H(M|z_{1:t}) = -\sum_{m \in M} p(m|z_{1:t}) \log p(m|z_{1:t})
\]

\[
\overline{H}(M|Z_{t+1}, z_{1:t}) = \int_{\hat{z} \in Z} p(\hat{z}|z_{1:t}) H(M|Z_{t+1} = \hat{z}, z_{1:t}) d\hat{z}
\]

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

\[
p(m[i]|\hat{z}_{t+1}^{[k]}, z_{1:t}) = \frac{p(\hat{z}_{t+1}^{[k]}|m[i])p(m[i]|z_{1:t})}{p(\hat{z}_{t+1}^{[k]}|z_{1:t})}
\]
entropy does not depend on the realization of future measurements, but it is an average over them. For all points in the perception field of the \( k \)-th sensor beam at the current robot location, \( \mathcal{I}_{t+1}^{[k]} \), the MI can be written as

\[
I^{[i]} = h(m^{[i]}) - \overline{H}(m^{[i]})
\]  

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 Figure 6 (right), an estimated MI map during an exploration experiment in the Cave environment [16] is depicted. In practice, at each time step, the map is initialized with the current map entropy, \( H(M|z_{1:t}) \), and for all map points inside the current perception field the estimated map conditional entropy values are 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.
4.2 Decision making

The resulting MI map shows the expectation for uncertainty reduction in the map at each place. In order 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 [38] 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 $\mathcal{A} = \{a^{[i]}\}_{i=1}^{n_a}$. We define the utility function as the difference between the total expected information gain predicted at the macro-action $a$, $U_I(a)$, and the corresponding path length from the current robot pose to the same macro-action, $U_d(a)$, as follows

$$U_I(a) \triangleq \sum_{k=1}^{n_a} \sum_{i \in \mathcal{I}[k]} I_i^{[k]}(a)$$

$$U(a) \triangleq \alpha U_I(a) - U_d(a)$$

where $\alpha$ is a factor to relate information gain to the cost of motion. The optimal action will thus be the one that maximizes the expected utility function. Therefore,

$$a^* = \arg \max_{a \in \mathcal{A}} E[U(a)]$$

The optimal action $a^*$ 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 literature [13, 40]. It lacks active searching for explicit loop closing actions, however, there are three main motivations to use the proposed utility function:

- For the information gain term, there is a guaranteed lower-bound because of the submodularity of MI.
- 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 Results and Discussion

In this section we present results from exploration experiments in the Intel map. The experiments include comparison among the original nearest frontier (NF) [46], the natural extension of NF with a GPOM representation (GPNF) [11], and the proposed MI-based (GPMI) exploration approaches. NF 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. Details about the compared methods are described in Table 3.
Figure 7: The constructed environment for exploration experiments using the binary map of obstacles from the Intel dataset.

Table 3: The compared exploration methods and their corresponding attributes.

|               | NF   | GPNF | GPMI |
|---------------|------|------|------|
| SLAM Mapping  | Pose SLAM | Pose SLAM | Pose SLAM |
| Frontiers     | binary | probabilistic | probabilistic |
| Utility Planner | path length | path length | MI+path length |
|               | $A^*$ | $A^*$ | $A^*$ |

5.1 Experiments setup

The environment is constructed using a binary map of obstacles and is shown in Figure 7. The simulated robot is equipped with odometric and laser range-finder sensors in order to provide the required sensory inputs for Pose SLAM. The odometric and laser range-finder sensors noise covariances were set to $\Sigma_u = \text{diag}(0.1 \text{ diag}, 0.1 \text{ diag}, 0.0026 \text{ diag})^2$ and $\Sigma_y = \text{diag}(0.03 \text{ diag}, 0.03 \text{ diag}, 0.0013 \text{ diag})^2$, respectively. Laser beams were simulated through ray-casting operation over the ground truth map using the true robot pose. In all the presented results, Pose SLAM [17] is included as the backbone to provide localization data together with number of closed loops and total information gain. Additionally, Pose SLAM parameters were set and fixed regardless of the exploration method. The localization mean square error (MSE) was 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 [41], frontier maps, and MI maps computations are listed in Table 4. The sensitivity of the parameters in Table 4 is not high and slight variations of them ($\sim 10\%$) do not affect the results presented in this article.
Table 4: Parameters for frontier and MI maps computations.

| Parameter                                      | Symbol | Value  |
|------------------------------------------------|--------|--------|
| 1) Beam-based mixture measurement model:       |        |        |
| Hit std                                        | $\sigma_{hit}$ | 0.03 m |
| Short decay                                    | $\lambda_{short}$ | 0.2 m |
| Max range                                      | $r_{max}$ | 4.00 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                     | $\beta$ | 3.0 |
| Logistic regression weight                     | $\gamma$ | 10.0 |
| 3) MI map and utility function:                |        |        |
| No. of sensor beams over 360 deg               | $n_z$  | 133 |
| Numerical integration resolution               | $s_z$  | 10/3 m$^{-1}$ |
| Information gain factor                        | $\alpha$ | 0.1 |
| Occupied probability threshold                 | $p_o$  | 0.65 |
| Unoccupied probability threshold               | $p_f$  | 0.35 |

The implementation has been developed in MATLAB and GP computations have been implemented by modifying the open source GP library in [34]. 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 smart mechanism has been adopted to address the problem. The cumulative relative entropy by summing the computed Jensen-Shannon divergence [26] can detect such map drifts. The method was first introduced in [11] and incorporated into the GP mapping methods in this article. For the frontier map computed using Equation (15), any point with probability greater than 0.6 was considered valid.

5.2 Exploration results in the Intel map

We compare the aforementioned techniques using nine different criteria in Table 5 and Figure 8. The figures are averaged over 10 independent run using the same setup and parameters. Frontier clusters with size larger than 14 cells (points) are considered valid in all the compared techniques, with a maximum number of 20 clusters. These conditions lead to avoiding excessive search and fluctuation in a small area and exploring the entire map faster.

Due to the defined utility function, the most significant part of the results is related to the map entropy rate in which the negative value means the map entropy has been reduced at each step. In NF and GPNF there is no prediction step regarding map entropy reduction, and therefore, the results are purely based on chance and structural shape of the environment. The robot only moves towards the nearest frontier and does not consider any form of uncertainty reduction which, also in turn, explains the lower map quality. GPNF and GPMI exploit I-GPOM2 for mapping, exploration, and planning. GP-based methods handle sparse sensor measurements by learning the structural dependencies (spatial correlation) present in the environment. The significant increase in the map entropy reduction rate is due to this fact and can be seen in Table 5. GPMI
maximizes the MI between the map and future measurements resulting in non-myopic decision making and the fastest map entropy reduction rate. The MI map is one-step look-ahead and the prediction is performed once using the corresponding macro-action. Therefore, for every macro-action there is one MI surface that is assumed to be fixed during the planning time – i.e., the robot executes an open-loop control strategy to reach the target. Figure 9 illustrates the indoor exploration results with the GPMI method in the Intel dataset where the sparse observations from a number of rooms and the hallway in the middle of the map sets a challenging test for the compared techniques.

5.3 Statistical analysis of the results

In order to interpret the results in a meaningful way, we conduct a statistical significance test using two-sample t-test at the 0.05 significance level. Table 6 shows the statistical significance test results and can be used together with the graphs and tables previously presented to interpret the results. GPMI outperforms NF and GPNF in terms of map entropy rate, AUC, and number of steps required to complete the map with higher computational time than NF as anticipated and yet, interestingly, not higher than GPNF. Although results from Table 5 give the impression that GPMI has the highest travel distance, the statistical analysis from the sample runs indicates that all three techniques are statistically close. In comparison with NF, GPMI maintains the robot well-localized. 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
Table 5: Numerical comparison of different exploration strategies in the Intel dataset (averaged over 10 experiments, mean ± standard error).

| Environment size: 35 diag × 35 diag; map resolution: 0.135 diag; minimum frontier cluster size: 14 |
|---------------------------------------------------------------|
| Travel distance (m)         | 437.23 ± 17.83 | 412.81 ± 21.87 | 534.93 ± 67.23 |
| Exploration time (min)      | 278.10 ± 13.88 | 371.76 ± 9.53  | 473.82 ± 57.55 |
| Map entropy rate (NATS/step)| -5.2065 ± 0.2634 | -6.4805 ± 0.3452 | -10.1443 ± 0.9757 |
| AUC – I-GPOM2               | 0.9957 ± 1.4033e-04 | 0.9956 ± 1.1902e-04 | 0.9962 ± 1.0169e-04 |
| AUC – OGM                   | 0.9722 ± 6.3927e-04 | 0.9631 ± 5.3793e-04 | 0.9641 ± 8.1962e-04 |
| Localization MSE (m)       | 0.1011 ± 0.0314  | 0.0408 ± 0.0079  | 0.0254 ± 0.0063  |
| Total information gain (NATS)| 306.6302 ± 20.5862 | 275.3049 ± 23.1657 | 412.5608 ± 69.7496 |
| Number of closed loops      | 70.10 ± 4.96     | 63.00 ± 5.59     | 96.70 ± 16.88    |
| Number of steps             | 87.10 ± 4.48     | 67.50 ± 3.52     | 43.90 ± 3.72     |

Table 6: Statistical significance test using two-sample t-test; $p \leq 0.05$. The ✓ sign shows the null hypothesis is rejected and the results are not from the same distribution – i.e., similar mean values. Accordingly, the × indicates that the null hypothesis is not rejected.

| Null hypothesis:                | $H_{GPNF} > H_{NF}$ | $H_{GPMI} > H_{NF}$ | $H_{GPMI} > H_{GPNF}$ |
|---------------------------------|---------------------|---------------------|-----------------------|
| Travel distance (m)             | ×                   | ×                   | ×                     |
| Exploration time (min)          | ✓                   | ✓                   | ×                     |
| Map entropy rate (NATS/step)    | ✓                   | ✓                   | ✓                     |
| AUC – I-GPOM2                   | ×                   | ✓                   | ✓                     |
| AUC – OGM                       | ✓                   | ✓                   | ×                     |
| Localization MSE (m)            | ×                   | ✓                   | ×                     |
| Total information gain (NATS)   | ×                   | ×                   | ×                     |
| Number of closed loops          | ✓                   | ✓                   | ✓                     |
| Number of steps                 | ✓                   | ✓                   | ✓                     |

improve the localization accuracy.

5.4 Computational complexity

The computational cost of GPs in big-$O$ notation is $O(n_t^3)$, given the need to invert a matrix of the size of training data $n_t$. BCM scales linearly with the number of map points. 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 $O(n_p^2n_zs_z)$.

A more sophisticated approximation approach can reduce the computational complexity further. The fully independent training conditional (FITC) [37] based on inducing conditionals suggests an $O(n_t^2n_i^2)$ upper bound where $n_i$ is the number of inducing points. More recently, in [15], the GP computation upper bound has been reduced to $O(n_i^3)$ which brings more flexibility in increasing the number of inducing points. This is a fitting complement to the incremental nature of the proposed mapping methodology in pursuit of manageable mapping options for real-time experiments.
Figure 9: MI-based exploration in the Intel map derived from the Intel dataset. (a) Pose SLAM map, (b) COM (I-GPOM2), (c) equivalent OGMs computed at the end of the experiment (d) corresponding entropy map of the COM (NATS). In (a), dotted (red) curves are the robot path and connecting lines (green) indicate loop-closures. The continuous occupancy map show 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.

6 Conclusion

We studied the problem of autonomous mapping and exploration for a range-sensing mobile robot using Gaussian processes. We formulated the GP occupancy mapping problem and presented algorithmic imple-
mentations of the solution using both batch and incremental techniques. In particular, we showed that the accuracy of our incremental mapping technique is comparable to that of the batch method and is faster. The proposed exploration strategy is based on map prediction using a forward sensor model and MI calculation. It is based on learning spatial correlation of map points with incremental GP-based regression from sparse range measurements, and computing MI surfaces estimation from the map posterior and conditional entropy. We used a statistical significance test for a meaningful interpretation of the exploration results.

When accurate sensors with large coverage relative to the environment are available, existing SLAM techniques can produce reliable localization without the need for active loop-closure detection. MI-based utility function proposed in this article is suitable for decision making in such scenarios. The more general form of this problem known as active SLAM requires active search for loop-closures in order 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. Integration of loop-closure detection in the utility function using a computationally tractable strategy will be the focus of our future work.

References

[1] Francesco Amigoni and Vincenzo Caglioti. An information-based exploration strategy for environment mapping with mobile robots. *Robot. Auton. Syst.*, 58(5):684–699, 2010. 3, 5

[2] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, MA, 1995. 15

[3] Frederic Bourgault, Alexei A Makarenko, Stefan B Williams, Ben Grocholsky, and Hugh F Durrant-Whyte. Information based adaptive robotic exploration. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, volume 1, pages 540–545. IEEE, 2002. 3, 5

[4] Luca Carlone, Jingjing Du, M Kaouk Ng, Basilio Bona, and Marina Indri. An application of kullback-leibler divergence to active slam and exploration with particle filters. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 287–293. IEEE, 2010. 3, 5

[5] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 1991. 15

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

[7] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006. 10

[8] Hans Jacob S Feder, John J Leonard, and Christopher M Smith. Adaptive mobile robot navigation and mapping. *The Int. J. Robot. Res.*, 18(7):650–668, 1999. 3, 5

[9] Maani Ghaffari Jadidi, Jaime Valls Miro, Rafael Valencia Carreño, Juan Andrade-Cetto, and Gamini Dissanayake. Exploration using an information-based reaction-diffusion process. In *Australasian Conf. Robot. Automat.*, 2013. 6
[10] Maani Ghaffari Jadidi, Jaime Valls Miro, Rafael Valencia Carreño, Juan Andrade-Cetto, and Gamini Dissanayake. Exploration in information distribution maps. In RSS Workshop on Robotic Exploration, Monitoring, and Information Content, 2013. 6, 7

[11] Maani Ghaffari Jadidi, Jaime Valls Miro, Rafael Valencia, and Juan Andrade-Cetto. Exploration on continuous gaussian process frontier maps. In Proc. IEEE Int. Conf. Robot. Automat., pages 6077–6082, 2014. 3, 6, 7, 17, 19

[12] Maani Ghaffari Jadidi, Jaime Valls Miro, and Gamini Dissanayake. Mutual information-based exploration on continuous occupancy maps. In Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., pages 6086–6092, 2015. 3

[13] H.H. González-Banos and J.C. Latombe. Navigation strategies for exploring indoor environments. The Int. J. Robot. Res., 21(10-11):829–848, 2002. 17

[14] Ruijie He, Emma Brunskill, and Nicholas Roy. Puma: Planning under uncertainty with macro-actions. In AAAI, 2010. 3

[15] James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. arXiv preprint arXiv:1309.6835, 2013. 21

[16] A. Howard and N. Roy. The robotics data set repository (Radish). http://radish.sourceforge.net, 2003. 4, 16

[17] V. Ila, J.M. Porta, and J. Andrade-Cetto. Information-based compact Pose SLAM. IEEE Trans. Robot., 26(1):78–93, 2010. 3, 5, 18

[18] M. Juliá, A. Gil, and O. Reinoso. A comparison of path planning strategies for autonomous exploration and mapping of unknown environments. Auton. Robot., pages 1–18, 2012. 3

[19] Brian J. Julian, Sertac Karaman, and Daniela Rus. On mutual information-based control of range sensing robots for mapping applications. The Int. J. Robot. Res., 33(10):1375–1392, 2014. 6, 15

[20] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. Artificial intelligence, 101(1):99–134, 1998. 15

[21] Soohwan Kim and Jonghyuk Kim. Building occupancy maps with a mixture of gaussian processes. In Proc. IEEE Int. Conf. Robot. Automat., pages 4756–4761. IEEE, 2012. 6

[22] Soohwan Kim and Jonghyuk Kim. Continuous occupancy maps using overlapping local gaussian processes. In Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., pages 4709–4714. IEEE, 2013. 3, 11

[23] Soohwan Kim and Jonghyuk Kim. Occupancy mapping and surface reconstruction using local gaussian processes with kinect sensors. IEEE Transactions on Cybernetics, 43(5):1335–1346, 2013. 7

[24] Soohwan Kim and Jonghyuk Kim. Gpmap: A unified framework for robotic mapping based on sparse gaussian processes. In Field and Service Robotics, pages 319–332. Springer, 2015. 7
[25] Andreas Krause and Carlos Guestrin. Near-optimal value of information in graphical models. In Conference on Uncertainty in Artificial Intelligence (UAI), July 2005. 3, 15

[26] J. Lin. Divergence measures based on the Shannon entropy. IEEE Trans. Information Theory, 37(1):145–151, 1991. 19

[27] A.A. Makarenko, S.B. Williams, F. Bourgault, and H.F. Durrant-Whyte. An experiment in integrated exploration. In Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., volume 1, pages 534–539, 2002. 5, 15

[28] Hans P Moravec and Alberto Elfes. High resolution maps from wide angle sonar. In Robotics and Automation. Proceedings. 1985 IEEE International Conference on, volume 2, pages 116–121. IEEE, 1985. 3

[29] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functionsi. Mathematical Programming, 14(1):265–294, 1978. 3, 15

[30] Simon O’Callaghan and Fabio Ramos. Continuous occupancy mapping with integral kernels. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 1494–1500, 2011. 11

[31] Simon O’Callaghan and Fabio Ramos. Gaussian process occupancy maps. The Int. J. Robot. Res., 31(1):42–62, 2012. 3, 6, 11

[32] Simon O’Callaghan, Fabio Ramos, and Hugh Durrant-Whyte. Contextual occupancy maps using gaussian processes. In Proc. IEEE Int. Conf. Robot. Automat., pages 1054–1060. IEEE, 2009. 6

[33] Fabio Ramos and Lionel Ott. Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent. In Robotics: Science and Systems, 2015. 6

[34] C.E. Rasmussen and C.K.I. Williams. Gaussian processes for machine learning, volume 1. MIT press, 2006. 6, 19

[35] Robert Sim and James J Little. Autonomous vision-based exploration and mapping using hybrid maps and rao-blackwellised particle filters. In Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, pages 2082–2089. IEEE, 2006. 15

[36] Robert Sim and Nicholas Roy. Global a-optimal robot exploration in slam. In Proc. IEEE Int. Conf. Robot. Automat., pages 661–666. IEEE, 2005. 5

[37] Ed Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In Advances in Neural Information Processing Systems 18, pages 1257–1264, 2006. 21

[38] Cyrill Stachniss, Giorgio Grisetti, and Wolfram Burgard. Information gain-based exploration using rao-blackwellized particle filters. In Robotics: Science and Systems, volume 2, 2005. 3, 5, 15, 17

[39] M.L. Stein. Interpolation of spatial data: some theory for kriging. Springer Verlag, 1999. 7

[40] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg. 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, 2003. 17
[41] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*, volume 1. MIT press, 2005. 3, 18

[42] V. Tresp. A Bayesian committee machine. *Neural Computation*, 12(11):2719–2741, 2000. 6, 10

[43] R. Valencia, J. Valls Miro, G. Dissanayake, and J. Andrade-Cetto. Active Pose SLAM. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 1885–1891, 2012. 5, 15

[44] R. Valencia, M. Morta, J. Andrade-Cetto, and J.M. Porta. Planning reliable paths with pose slam. *Robotics, IEEE Transactions on*, 29(4):1050–1059, 2013. ISSN 1552-3098. doi: 10.1109/TRO.2013.2257577. 5

[45] Joan Vallvé and Juan Andrade-Cetto. Dense entropy decrease estimation for mobile robot exploration. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 6083–6089. IEEE, 2014. 5

[46] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Int. Sym. Comput. Intell. Robot. Automat.*, pages 146–151, 1997. 5, 17

[47] Kwangjin Yang, Seng Keat Gan, and Salah Sukkarieh. A Gaussian process-based RRT planner for the exploration of an unknown and cluttered environment with a UAV. *Advanced Robotics*, 27(6):431–443, 2013. 6