Occupancy Map Building through Bayesian Exploration

Gilad Francis, Lionel Ott, Roman Marchant and Fabio Ramos

Abstract
We propose a novel holistic approach for safe autonomous exploration and map building based on constrained Bayesian optimisation. This method finds optimal continuous paths instead of discrete sensing locations that inherently satisfy motion and safety constraints. Evaluating both the objective and constraints functions requires forward simulation of expected observations. As such evaluations are costly, the Bayesian optimiser proposes only paths which are likely to yield optimal results and satisfy the constraints with high confidence. By balancing the reward and risk associated with each path, the optimiser minimises the number of expensive function evaluations. We demonstrate the effectiveness of our approach in a series of experiments both in simulation and with a real ground robot and provide comparisons to other exploration techniques. Evidently, each method has its specific favourable conditions, where it outperforms all other techniques. Yet, by reasoning on the usefulness of the entire path instead of its end point, our method provides a robust and consistent performance through all tests and performs better than or as good as the other leading methods.

Keywords
Robotic exploration, Bayesian optimisation

1 Introduction
Autonomous exploration is a challenging dynamic decision-making process, where the goal is to build a representation of an initially unknown environment. While exploring, the robot determines its own position and decides where to move next based on its task. Ideally, these decisions correspond to continuous trajectories which determine the goal pose of the robot and a path leading to that point. Such a path would maximise the robot’s objective, while ensuring safety. However, given the dimensionality and shape of the search space imposed by the motion constraints, a closed-form solution to the general exploration problem is intractable. The numerous exploration techniques available in the literature provide different approaches for making a decision on which path to follow, each with its own strategy for dealing with the uncertainty of the model and solution horizon. Risk and safety are typically addressed during execution of the chosen path, and not as an integral part of the exploration decision-making process.

In this paper, we present a novel framework for autonomous exploration over continuous paths using constrained Bayesian optimisation (BO) (Gelbart et al. 2014), which we term constrained Bayesian exploration (CBE). A-priori, the exploration objective and constraints functions are unknown, i.e. have no closed-form expression. Knowledge about any of these functions is obtained solely from noisy observations, typically using forward simulation. A naive approach for optimising path decision in such a case is to employ an exhaustive search. Of course, such a process is computationally infeasible due to the size of the search space and cost associated with evaluating the reward and constraints functions over entire paths. BO uses a completely different approach, which turns autonomous exploration into an active learning process over the continuous search space. Instead of exhaustive sampling, BO learns probabilistic surrogate models for the expensive-to-evaluate reward and constraints functions (Brochu et al. 2010). A simple and cheap heuristic function, called acquisition function, guides an efficient sampling schedule based on the posterior mean and variance of the surrogate models. The acquisition function balances the exploration-exploitation trade-off which guarantees convergences to the optimum while ensuring probabilistic completeness of the objective and constraints functions models.
The main contributions of this paper are; First, we formulate the problem of building an occupancy map of an unknown environment as an optimisation problem in a constrained continuous action domain. Second, we develop CBE, an innovative method to solve this problem, i.e. optimise path decision whilst keeping the robot safe and within its dynamic constraints.

The remainder of this paper is organised as follows. Section 2 surveys the work related to autonomous exploration. Section 3 provides background on combined exploration and map building. Section 4 gives an introduction of the basic building blocks of constrained Bayesian exploration and details the algorithms behind CBE. Experimental results and analysis for various scenarios are shown in Section 5. Finally, Section 6 draws conclusions on the proposed method.

2 Related Work

Autonomous exploration can be seen as an active learning process aimed at minimising uncertainty and producing high-fidelity maps (Makarenko et al. 2002; Stachniss 2009). Exploration requires solving simultaneously mapping, path planning and localisation. Due to its complexity, existing research has mainly focused on solving a relaxed form of this problem, by either decoupling processes or by limiting the solution search space.

The plethora of autonomous exploration methods is categorically divided into two branches; frontier-driven and information-theoretic. A quantitative comparison between the various exploration algorithms is presented in Juliá et al. (2012).

The key concept in any frontier-based exploration is moving towards the edges of the known space, i.e. the boundary between free space and unmapped regions (Yamauchi 1997). In its simplest form, after identifying and clustering frontiers, the robot moves towards the closest one. Other authors suggest various utility functions to prioritise candidate frontier locations. González-Baños and Latombe (2002) used expected information gain at the frontier and travelling cost. Basilico and Amigoni (2011) incorporated the overlap with known space as a measure for self-localisation. Extensions for 3D autonomous exploration have also been suggested by various authors (Dornhege and Kleiner 2013; Shen et al. 2012; Shade and Newman 2011).

Information driven exploration strategies minimise a utility associated with the uncertainty of the map. Early work dealt with only finding the next best view (NBV), which is the discrete location that will have the greatest effect on the utility function. Whate and Ferrie (1997) proposed minimising the entropy of the map. Mutual information (MI) has also been suggested as a measure for the predicted reduction of map uncertainty (Elfe 1996; Bourgault et al. 2002; Julian et al. 2014) suggested MI to encode geometric dependencies and drive exploration into unexplored regions in a similar fashion to frontier-based approaches. Makarenko et al. (2002) proposed an integrated exploration that combines the goal functions of map and localisation uncertainties with the cost of navigation, balancing exploration with simultaneous localisation and mapping (SLAM) loop-closures. Tovar et al. (2009) extended this technique by selecting several observation points using a tree search. Yet, decision on the path passing through these points is still not part of the optimisation process. Stachniss et al. (2005) used a particle filter to calculate the expected information gain of an action. However, this formulation still defines point actions of either loop-closure or exploration. A method to generate a path based on information potential fields is proposed by Vallvé and Andrade-Cetto (2015). The path is generated by applying a grid-step gradient on the potential fields, hence the resulting path does not necessarily complies with the robot kinematic restriction.

Several non-myopic exploration methods have emerged in recent years. These methods treat exploration as a sequential decision process. Yang et al. (2013) used a rapidly-exploring random tree (RRT) planner with Gaussian process occupancy map to generate a safe path that minimises MI. Similarly to CBE, the RRT planner does not set a goal point but rather explores promising paths. The difference in algorithms lies in the optimisation process. While RRT uses predefined valid branches for its tree, CBE optimises path selection over the continuous domain. Furthermore, CBE learns the path constraints, which makes it a more flexible algorithm. Charrow et al. (2015) combined both frontier and information-theoretic approaches. Global goal candidates are produced by identifying frontiers. A coarse path to each candidate is generated using local motion primitives that satisfy the kinematic envelope of the robot. After assessing the information gain of all candidates, the best path is refined by a local gradient-based optimiser. While this method optimises control inputs in continuous space, the search is limited to a single promising path. CBE, on the other hand, does not define a goal point, and its optimisation is done on the entire domain of controls. Kollar and Roy (2008) proposed an exploration procedure that maximises map coverage, by choosing a set of observation points that the robot trajectory must pass through. The executed path minimises the errors introduced by the robot motion. The control policy is implemented by a support vector machine (SVM) classifier trained off-line. Another exploration approach was introduced by Marchant and Ramos (2014). In this case, Bayesian optimisation was used to learn and optimise a utility function along continuous paths. They employed two instantiations of BO,
one for the probabilistic model of the utility and another to select a continuous informative path. However, their optimisation process does not consider any motion or safety constraints, which are learned and incorporated in the CBE framework. In a more recent work, Marchant et al. (2014) developed a sequential Bayesian optimisation method within a partially observable Markov decision process (POMDP) framework. They used Monte Carlo tree search (MCTS) to approximate the unconstrained solution for a spatio-temporal monitoring problem. Martinez-Cantin et al. (2007) and more recently Martinez-Cantin et al. (2009) utilised Bayesian optimisation to find control policies that minimise the state error of a robot and landmarks. While they used a different cost function, their method resembles the approach taken in this work. However, CBE extends this method by incorporating and learning unknown constraints during the optimisation. Lauri and Ritala (2015) used a POMDP with a MI objective to plan exploration paths with a fixed horizon using tree search methods to optimise the exploration policy. Their method relies on a Monte-Carlo (MC) approximation for the MI objective forward simulation. CBE, on the other hand, uses BO to efficiently manage the objective and constraints functions sampling. Another POMDP continuous-domain planning technique was presented by Indelman et al. (2015). They treated the exploration as an optimisation of the expected cost over several look-ahead steps. The cost at each step is inferred from the joint probability distribution of the robot and environment state. As the expected cost has a closed form expression, the authors use gradient descent to locally optimise the policy selection. A similar approach was taken by Rafieisakhaei et al. (2016) which defined a penalised cost function based on the maximum-a-posteriori (MAP) state estimate and control effort penalties. Our method takes a different approach to optimisation. Instead of using a closed-form expression for the objective, CBE learns it from samples, which results in a more flexible solution. Therefore, there is no need to define an expression for the objective and constraints. Rather the non-parametric structure captures our belief about the cost function. Coupling that with the BO framework, provides better guarantees that a solution will converge to the global optimum.

In summary, in this paper we take a more holistic approach to exploration. Ideally, one would like to select the path that yields the highest reward. However, evaluating path reward is expensive and as such, simple global optimisation strategies are rendered impractical. Therefore, most exploration techniques break this problem into two sub-problems; finding the next observation point(s) and selecting a path through these points. By contrast, our method uses a modified version of BO that finds a solution for these two sub-problems at the same time. The reward function and any associated motion constraints, such as turn rate limits or obstacles, are treated as functions which are learned by the optimiser. The output of the optimisation procedure is a path that maximises the reward without violating any of the constraints.

3 Exploration and Map Building

In this section we formally describe the problem of safe autonomous exploration for building occupancy maps. We start by describing the map representation before formulating the exploration process as an optimisation problem over the robot’s action space.

3.1 Occupancy Grid Maps

The aim of autonomous exploration is to map an unknown environment. Formally, mapping is an inference process where given a set of observations, \( z \), taken at known poses, \( x \), the posterior distribution over maps, \( m \), is given by \( p(m|x,z) \). Using grid maps, the complexity of calculating the posterior can be reduced (Thrun et al. 2005). In a grid map, the 2D world is discretised into cells, where each grid cell, \( m_i \), is an independent Bernoulli random variable. This assumption simplifies calculations since the map posterior is now a product of the individual cells:

\[
p(m) = \prod_i p(m_i). \tag{1}
\]

It is worth noting that while the exploration algorithm drives the process of map building, updating the map with new observations is controlled by a separate and external routine; e.g. gMapping (Grisetti et al. 2007). As such, the use of occupancy grid map is not critical for CBE, and other techniques such as Gaussian process occupancy maps (GPOM) (O’Callaghan and Ramos 2012) or Hilbert maps (Ramos and Ott 2015) can be used. However, working with a different type of occupancy map requires changing the calculation of the reward estimation discussed in Section 4.4.

3.2 Exploration as an Optimisation Problem

An optimal path is a series of control inputs \( u^* = (u_1, u_2, \ldots, u_n) \) that minimises a desired objective function, \( f \), over a valid region \( C \)

\[
u^* = \arg \min_u f(m,u) \text{ s.t. } u \in C. \tag{2}
\]

In most cases, there is no closed-form expression for \( f \) or \( C \). Rather, both are a-priori unknown and thus can only be estimated from sparse, expensive-to-evaluate and potentially noisy observations (samples). Accordingly, stochastic models are well suited to represent both \( f \) and \( C \). However, using probabilistic models for \( f \) and \( C \) forces
changes in the formulation of the optimisation problem: (i) optimisation is performed on the expected value of the objective function; (ii) the constraints are estimated using confidence bounds, $\delta$, indicating there is high probability that the constraint is met \cite{Brochu2010}, which transform Eq. \eqref{eq:2} into:

$$
\mathbf{u}^* = \arg \min_{\mathbf{u}} \mathbb{E}[f(m, \mathbf{u})] \text{ s.t. } \Pr(C(\mathbf{u})) > 1 - \delta. \quad (3)
$$

However, solving Eq. \eqref{eq:3} in a continuous action space, $\mathbf{u}$, is computationally infeasible. A common approach, used by most information-theoretic methods, to reduce complexity is to search for NBV by optimising in pose space. The path planning process is divided into two separate sub-processes. The goal of the first sub-process is to define a set of discretised view points. Each point is the spatial local extremum of the objective function. The second sub-process plans a path from the current location of the robot to its next observation pose and is determined according to obstacles and robot’s kinematic constraints \cite{Makarenko2002, Kollar2008}. Although the discrete view points approach simplifies the optimisation process, the resulting path is suboptimal. The main drawback of this approach is that it only considers a limited set of points and thus disregards the potential gains (or the lack thereof) along the entire path. A less obvious, yet significant disadvantage of discrete optimisation, stems from the fact that the expected cost of the resulting path are not an integral part of the view point selection. Penalty heuristics, e.g. distance to goal point, are typically incorporated in the objective function in order to encode cost, yet their limited form underestimates the real cost imposed by motion and safety constraints. As a simple example, consider the case in which the next view point is located close to the robot but is separated by obstacles. While a valid path to that point might exist, it is less desirable due to excessive cost.

The following section describes our approach for solving Eq. \eqref{eq:3} by optimising the utility of the entire path while minimising the risk of violating any constraints.

4 Constrained Bayesian Exploration (CBE)

Finding the best path in a continuous space requires optimising a reward function evaluated for any given trajectory. However, computing such a reward for the entire space of paths is computationally infeasible. Furthermore, obstacles and the robot’s kinematic envelope impose constraints that might not have a closed form expression or are not known \textit{a-priori}. BO provides a strategy to learn the reward function and constraints while searching for the valid extremum.

BO guides the optimisation process. Given a sparse training set, BO builds models of the reward function and constraints using Gaussian processes (GPs). With these models, BO identifies promising trajectories that correspond to the optimal path with high probability. Constraints are handled in a statistical manner, with BO balancing risk and rewards.

A schematic overview of CBE is illustrated in Fig. \ref{fig:1}. As with any autonomous exploration technique, the expected output of our method is a path that updates our belief about the map. As shown, constrained BO exploration is an iterative process of optimisation and learning. Using the current map as an input, BO explores the solution space by sampling promising path candidates. The samples update the surrogate GP models for the reward and constraints, which is followed by the next BO suggestion. This process continues until resources are exhausted and the optimal path is then selected and executed.

In the following subsections we review the building blocks of CBE. Section 4.1 provides a short introduction to GP regression and classification, the engine behind the BO surrogate models. Descriptions of the BO method and the constrained BO framework are given in sections 4.2 and 4.3 respectively. In section 4.4 the CBE method is described in detail. Finally, a modification of the CBE algorithm to accommodate uncertainty in the robot pose is presented in 4.5.

4.1 Gaussian Processes

4.1.1 GP regression GPs are an elegant nonparametric regression technique that places a multivariate Gaussian
distribution over the space of functions [Rasmussen and Williams 2006]. A GP is completely defined by a mean function \( m(x) \) and a covariance or kernel function \( k(x, x') \) given an input space \( x \in \mathbb{R}^D \). Both functions are user-defined and encode prior belief about the properties of the regression problem.

A GP builds a model of the unknown underlying function \( f(x) \) based on a set of noisy observations \( D = \{x_i, y_i\}_{i=1}^N \). Here, \( x_i \in \mathbb{R}^D \) is the location of the \( i \)-th sample point and \( y_i \in \mathbb{R} \) is the corresponding target value. In addition, there is no direct access to \( f(x) \) and only noisy observations from \( f \) are available according to \( y = f(x) + \epsilon \), where the noise follows a Gaussian distribution \( \epsilon \sim \mathcal{N}(0, \sigma_n^2) \).

Conditioned on observations \( D \), inference at test point \( x^* \) corresponds to the predictive distribution for \( y^* = f(x^*) \):

\[
y^* \mid x^*, D = \mathcal{N}(\mu, \sigma^2),
\]

where,

\[
\begin{align*}
\mu &= K(x^*, X)[K(X, X) + \sigma_n^2 I]^{-1} y, \\
\sigma^2 &= K(x^*, x^*) - K(x^*, X)[K(X, X) + \sigma_n^2 I]^{-1} k(X, x^*).
\end{align*}
\]

Here \( X = \{x_i \in D\}_{i=1}^N \) and \( x^* \) is the test point. \( K(X, X) \) is the covariance matrix between all pairs of inputs in the training set. \( K(X, x^*) \) is the covariance matrix calculated between the test point, \( x^* \), and the observation set. Every element in the covariance matrix is calculated from the kernel function \( k(x, x') \) and its associated set of hyperparameters \( \theta \).

The choice of hyperparameters, \( \theta \), is critical for a successful GP model. The right value for the hyperparameters ensures that the model generalises well without over-fitting the data. The common practice for training a GP’s hyperparameters is to maximise the model log marginal likelihood

\[
\theta_{\text{optimal}} = \arg \max_{\theta} \log p(y|X, \theta).
\]

We refer the reader to an extensive discussion on GP hyperparameters training in [Rasmussen and Williams 2006].

4.1.2 GP Least Squares Classifier (GPC) GP Least Squares Classifier (GPC) is a simple and efficient classification method based on GP regression [Rasmussen and Williams 2006]. In binary classification, the observation targets \( y_i \) can be either \( \{-1, +1\} \) with a corresponding probability of \( p \) and \( 1-p \), respectively. With the GPC model, the squared error of a training data point is minimised by mapping the output of a GP regressor to the \([0, 1]\) interval to obtain the class probability for a test point, \( p(y^*|D, x^*) \). This is attained by post-processing the output of the regressor using a "squashing" sigmoid that confines the regression output to the interval \([0, 1]\). Using a cumulative Gaussian sigmoid, the class probability is computed as

\[
p(y^*|x^*, D, \theta) = \Phi \left( \frac{y^*(x^*) + \beta}{\sqrt{1 + \alpha^2 \sigma(x^*)^2}} \right),
\]

where \( y^* \) is either \(+1\) or \(-1\). \( \mu \) and \( \sigma \) are calculated from Eqs. (5) and (6), respectively. \( \Phi \) stands for the Gaussian cumulative distribution, \( \theta \) represents the hyperparameters of the GP regressor and \( \alpha \) and \( \beta \) are the "squashing" parameters of the classifier. Training the GPC hyperparameters \( \theta, \alpha, \beta \) is a two-step process. First, the regression hyperparameters, \( \theta \), are trained by maximising the log marginal likelihood as with conventional GP regression. Second, the squashing parameters, \( \alpha \) and \( \beta \) are trained by maximising the log predictive probability using a leave-one-out cross validation (LOOCV) [Rasmussen and Williams 2006]:

\[
L_{\text{LOOCV}} = \sum_{i=1}^{N} p(y^i|x, y_{-i}, \alpha, \beta)
\]

The \(-i\) subscript indicates that target \( y_i \) is excluded. The GPC framework provides closed-form expressions for \( p(y^i|x, y_{-i}, \alpha, \beta) \) and for the derivatives of \( L_{\text{LOOCV}} \) with regards to the hyperparameters, \( \alpha \) and \( \beta \), which facilitate an efficient training process.

4.2 Bayesian Optimisation

BO is a powerful global optimiser, which is most effective when the objective function does not have a closed-form expression, costly to evaluate and there is no access to derivative information. Given a limited set of noisy observations and prior beliefs about the properties of the objective function, BO exploits Bayes’ theorem to determine the most effective course of action [Brochu et al. 2010].

The building blocks of a Bayesian optimiser are the surrogate and acquisition functions. The surrogate function is the estimated model of the objective function we would like to optimise. It holds our current belief of the underlying function, which is inferred from observations and prior knowledge of its properties. Gaussian processes (GPs) are generally used for modelling the surrogate function due to their Bayesian non-parametric properties and analytical form. When modelled using a Gaussian process (GP), the surrogate function is represented by the posterior mean and variance.

The acquisition function guides the selection of new observation points to sample from the unknown objective
function. Based on the current model of the objective function, it provides a quantitative measure for the probability of finding the global extremum in a specific location. The Bayesian optimiser uses this measure as a utility proxy to select the next observation point. In essence, the original optimisation problem, Eq. (3), is transformed into an iterative optimisation process of the acquisition function:

$$x_{k+1}^u = \arg\min_{x^u} s(x^u). \quad (10)$$

The superscript $u$ is used for brevity and implies that the result of the optimisation process is a set of control inputs, $u$, as defined in Section 3 i.e. $x^u = u$.

Although this is still a non-convex optimisation problem, using an appropriate acquisition function makes the search for the extremum more efficient. A proper acquisition function balances the exploration-exploitation trade-off. Therefore, the optimisation process considers both where we believe the extremum lies and promising location in unexplored regions. Consequently, the number of expensive evaluations of the objective function is kept to a minimum. Brochu et al. (2010) list the most commonly used acquisition functions. A modified version of these acquisition functions for autonomous exploration will be used acquisition functions. A modified version of these functions takes the following analytic form:

1. **Expected Improvement (EI).** EI is defined as the expected difference from the true extremum. On its $k+1$-th iteration, the optimiser finds a location that maximises the expected difference from the true extremum, $f(x_{min}^u)$. In a minimisation problem, finding $f(x_{min}^u)$. EI is defined as follows:

$$EI(x^u) = \mathbb{E}\{f(x^u) - f(x_{min}^u)\}. \quad (11)$$

We follow a slightly modified version of EI that uses the predicted mean, $f_{min}$, which is inferred from the GP model (Gramacy and Lee 2011). Furthermore, we exploit the GP structure to produce a concise closed-form for EI:

$$EI(x^u) = \begin{cases} -\sigma(x^u)[Z\Phi(Z) + \phi(Z)] & \sigma(x^u) > 0 \\ 0 & \sigma(x^u) = 0 \end{cases} \quad (12)$$

Where $\phi$ and $\Phi$ represent the normal distribution PDF and CDF, respectively. $\sigma(x^u)$ is the standard deviation of the posterior distribution in $x^u$. $Z$ is given by

$$Z = \begin{cases} (f_{min}^u - \mu(x^u) - \zeta)/\sigma(x^u) & \sigma(x^u) > 0 \\ 0 & \sigma(x^u) = 0 \end{cases},$$

where $\zeta$ is a user-defined parameter that balances the exploration-exploitation trade-off.

### Algorithm 1: Bayesian Optimisation

**Input:** $f(x^u)$: Objective function. $s(x^u)$: Acquisition function. $f_{min}$: Current minimum.

**Output:** $x_{min}^u$ for $i = 1, 2, 3, \ldots$ do

1. Find: $x_{i}^u \leftarrow \arg\min x^u f(x_{i}^u)$
2. Sample objective function: $f_i \leftarrow f(x_{i}^u)$
3. Update GP model with new observation ($x_{i}^u, f_i$)
   - if $f_i < f_{min}$ then
     - $x_{min}^u \leftarrow x_{i}^u$
     - $f_{min} \leftarrow f_i$
   - end

end

2. **Lower Confidence Bound (LCB).** The LCB lacks the rigour of EI. However its user-defined parameter, $\kappa$, provides a simple mechanism to adjust the optimisation exploration-exploitation trade-off:

$$LCB(x^u) = \mu(x^u) - \kappa\sigma(x^u) \quad (13)$$

The pseudo code shown in Algorithm 1 outlines the typical steps performed by BO. In each iteration, a new sampling location, $x_i^u$, is found by minimising the acquisition function $s(x^u)$. BO evaluates the objective function, $f$, at $x_i^u$ and checks whether a new extremum has been found. In addition, the new observation, $f_i$, updates the surrogate (GP) model, which holds our belief of $f$. The updated models are then used on the next iteration of BO. By choosing an appropriate GP model and acquisition function, BO keeps the number of function evaluations low, leading to an efficient optimisation process.

A one dimensional example of BO is depicted in Fig. 2. The optimiser has no knowledge of the objective function (blue line) other than the noisy samples (red asterisks). A GP model is generated based on these observations. The model is accurate and confident around the sampling points, where the posterior mean (black dashed line) converges to the objective function values and its variance (grey shade) is low. Initially, the LCB acquisition function resembles the GP variance, which leads to an aggressive exploratory behaviour at the beginning of the optimisation. As the model becomes more confident, the optimiser focuses its search around the global minimum.

### 4.3 Constrained Bayesian Optimisation

Constrained BO handles optimisation with unknown constraints. Similarly to the optimisation process discussed in the previous section, our only knowledge of the underlying constraints stems from observations. Furthermore, the
objective function is undefined wherever the constraints are violated. To tackle the uncertainty in both objective function and constraints, we employ a constraint weighted acquisition function (Gelbart et al. 2014). Consequently, the optimisation balances the expected reward with the confidence in the constraint model and its associated risk.

In the literature, there are two different modifications to the basic BO acquisitions function relevant to our case. Gramacy and Lee (2011) propose a variant of EI called integrated expected conditional improvement (IECI). In its essence, IECI represents the marginal effect that a new observation will have on the overall uncertainty of the GP model regardless of its actual value and is defined as

\[
IECI(x^u) = \int \mathbb{E}(EI(x' | x^u)) c(x' | x^u) dx'
\]

(14)

To incorporate constraints, Gramacy and Lee integrated the conditional improvement under the probability density of the constraint function, \(c(x' | x^u)\). The main caveat of this method is its scalability. Calculating the integral over the expected conditional improvement requires heavy Monte Carlo sampling of the GP model. Hence, IECI is not a practical method for real-time problems. Furthermore, this method might not be suitable for most standard constrained optimisation problems since it assumes that the objective function can be sampled in regions where the constraint is violated.

The other modification to BO, which we use in this work, is the constraint weighted acquisition function proposed by Gelbart et al. (2014). The confidence in the validity of the solution scales the expected utility of the acquisition function. With independent constraints, a Constrained LCB (CLCB) is thus defined as:

\[
CLCB(x^u) = LCB(x^u) \prod_{k=1}^{K} \Pr(C_k(x^u) < 1 - \delta_k).
\]

(15)

Where \(\delta_k\) is a user defined constrained confidence bound over constraint \(k\).

In order to model the constraints \(C_k(u)\), we employ GPCs, \(g_k\), to provide an estimate for the likelihood constraint \(k\) is satisfied within the user defined confidence bounds \(\delta\):

\[
CLCB(x^u) = LCB(x^u) \prod_{k=1}^{K} \Pr(g_k(x^u) < 1 - \delta_k).
\]

(16)

Incorporating learned constraints complicates the optimisation algorithm as shown in pseudo code in Algorithm 2. Since the objective function is undefined in regions where the constraints are not satisfied, a preprocessing step finds feasible and valid regions. Within these regions, the optimiser finds the next observation using the utility function defined in Eq. (16). With every new observation point, the constraints are assessed and their respective GPC model is updated. The GP model for the goal function, on the other hand, is only updated when all constraints are met.

A one dimensional example for constrained BO is shown in Fig. 2. The unknown constraint is indicated by the area shaded in green, while its predictive probability is represented by the blue area. As with the regression of the objective function, the confidence in the constraints value, whether valid or invalid, is higher around observation points. As evident from Fig. 2 BO tries to evaluate points outside the constrained region, however this only updates the constraint model while the objective GP model is unchanged (hence uncertainty is high). With every observation, BO becomes more confident in the model of the objective function, borders of the constraint and the location of the global minimum.

4.4 Constrained Bayesian Exploration (CBE)

When planning a path for autonomous exploration, the objective is to acquire new knowledge and improve the existing map in an efficient manner. Yet, both robot and environment impose restrictions on the solution space. Any selected path should be safe, i.e. free of obstacles, and within the kinematic envelope of the robot. Constrained
Algorithm 2: Constrained Bayesian Optimisation

**Input:** $f(x^u)$: Objective function.
$s(x^u)$: Acquisition function.
$g_k(x^u)$: k-th constraint function.
$\delta_k$: k-th constraint tolerance.
$f_{\text{min}}^*$: Current minimum.

**Output:** $x_{\text{min}}^u$

for $i = 1, 2, 3, \ldots$ do
    feasible region $C$
    $C(x^u) = \prod_{k=1}^{K} \Pr(g_k(x^u) < 1 - \delta_k)$
    Find next sample point:
    $x_i^u \leftarrow \arg\min_{x^u \in C} s(x^u) \prod_{k=1}^{K} \Pr(g_k(x^u) < 1 - \delta_k)$
    if $x_i^u$ valid then
        Sample objective function: $f_i \leftarrow f(x_i^u)$
        Update GP model with new observation
        $(x_i^u, f_i)$
        if $f_i < f_{\text{min}}^*$: $x_{\text{min}}^u \leftarrow x_i^u$
        end
    end
    Update GPCs with new observation
end

BO is a very flexible tool to find such a solution. Safety and motion restrictions are treated as constraints, which are learnt while the optimiser searches for the global extremum among the possible paths.

CBE is an iterative process, of finding a promising sampling location, observing the unknown objective and constraint function values at that location before updating the model using the observed values. In the context of robotic exploration, sampling a location corresponds to testing a new path candidate. It is important to note that while CBE optimises the controls $u \in \mathbb{R}^D$, the objective function and constraints are observed along trajectories, $\xi(u) \in T$. While $T$ is a subspace of the robot’s configuration space, we avoid the explicit notation, since the exact structure of the configuration space is unknown. The pseudo code for assessing a new action, i.e. new path, is shown in Algorithm 3 and it categorically consists of two major parts: (i) Path Candidate Validity Assessment, (ii) Reward calculation.

### 4.4.1 Path Candidate Validity Assessment

The role of the path candidate validity assessment is to determine whether a path is valid or not. A valid path is safe from collisions with obstacles and within the kinematic capabilities of the robot.

There are two tests to assess the validity of a new path. First, the maximum curvature, $\kappa_{\text{max}}$, along the path is evaluated. If it exceeds a user-defined threshold, $\kappa_{\text{max}} > \delta_c$, then the path is considered invalid and no other tests are needed. Although this is a relativity simple constraint, it is implemented in the current minimum

\[
\text{safe} = \begin{cases} 
1 & m[\xi(u)] < \delta_{\text{safe}} \\
0 & \text{o.w.}
\end{cases}
\]

It is important to note again, that the result of Eq. (17) is used as a point observation in the generation of a stochastic model for the safety constraints. As this is a learned model, which is based solely on these observations, the exact implementation of the safety criteria is unimportant. Instead of using $m[\xi(u)] < \delta_{\text{safe}}$, the user can define a different test to assess path safety that better suits the robot and...
environment configuration. The simplest example changes the confidence safety threshold, $\delta_{safe}$, which will modify the risk-reward balance. More complicated methods may use different occupancy maps or include visibility test for dynamic obstacles.

In case the path is invalid, an additional post-processing step is taken in order to better define the valid solution space. The invalid path is expanded into a subset of derived valid and invalid paths, which are used to update the GP models. The added observations help reduce the uncertainty of the relatively sparse GP models. But more importantly, it provides the GP model with the boundaries between the valid and invalid space.

4.4.2 Reward calculation
In autonomous exploration, a commonly optimised objective function is information gain (IG). Information gain measures the reduction in the map entropy after observations are made

$$IG(u) = H(m) - H(m|u)$$

(18)

Here $H(m)$ is the entropy of the map and $H(m|u)$ is the entropy after a path, defined by a set of actions $u$, was executed. In order to evaluate the information gain along an entire path, it is necessary to emulate the expected laser observations along the path and their accumulated effect on the map’s statistical model. This is achieved by ray tracing the simulated beams originating from a robot moving on the path.

For an effective optimisation, we limit the action search space to be a parametrised trajectory originating from the robot’s current pose. In this work we use quadratic splines, however other forms of trajectory representation can be easily utilised. The limitation of parametrised trajectories is their expressivity, especially when constraints are present. The effect this limitation has on the optimisation process is more profound in areas that were explored by the robot before or around dead-ends. In such areas, the reward is similar in all valid paths. Meaning that with the limited choice of paths and under the imposed constraints, the robot can not resolve a viable decision that will pull it toward unexplored regions. To alleviate this problem, we introduce two penalties that affect the optimisation process only when the local differences are small:

- The first penalty provides a global context to the overall objective function. A coarse path is planned from the robot’s location to the nearest frontier. This path does not have to be traversable by the robot and it can violate safety or kinematic envelope constraints. However, it biases path selection toward a region where IG reasoning is feasible. We define a penalty value, $P_H(u)$, which is a function of the difference between the direction of the assessed path and the coarse path. Therefore, a path that develops in the opposite direction of the global coarse path will have higher penalty than a path that is oriented towards a similar direction.

- The second penalty is a function of the path length, $P_L$. The rationale is to penalise very short and longs paths. This will drive the robot forward, whilst preventing overly confident longer paths.

The additional penalties are added to the IG reward with corresponding weights, $W_1$ and $W_2$. These weights keep the penalties small compared to the typical IG utility:

$$IG_{Modified}(u) = IG(u) + W_1 \cdot P_H(u) + W_2 \cdot P_L(u)$$

(19)

Even with the simplest occupancy map representation, the forward projection model needed to estimate IG is expensive to evaluate. This is the main motivation for using BO: optimising decision making while keeping sampling low. Instead of optimising by explicitly calculating the forward simulation IG results, BO learns IG by sparsely sampling it and building an equivalent model. It then uses these models to infer the next sampling location. The efficiency of BO relies on the accuracy of the learned GP models. However, a high fidelity GP surrogate model requires a substantial number of function observations. To increase the number of sample points and fill in the gaps in the GP model, we notice that the IG along the path is a non-decreasing monotonic function. Since the robot motion along the path is a set of sequential observation points, the IG in any given point is the sum of accumulated IG of all previous observations and the contribution of the current observation

$$IG([u_1...u_{k+1}]) = IG([u_1...u_k]) + \delta IG(u_{k+1}|z_{k+1})$$

Thus, by evaluating the path reward sequentially, a denser GP model for the objective function can be generated at no additional computational cost.

4.4.3 CBE Algorithm

A pseudo code for CBE using constrained Bayesian optimisation is given in Algorithm 4. Fig. 3 provides a visual explanation of this process. The algorithm is divided into two parts. First, a training set consisting of a small number of paths, typically 20-50, is formed. The training set is shown in Fig. 4a) by the red (invalid) and cyan (valid) paths. As explained before, the algorithm tries to extract valid path segment from an invalid path as can be seen in Fig. 4b. These paths, valid as well as invalid, are used in the update of the GP and GPCs models, which serve as a prior model for the subsequent constrained BO stage.

The second part of Algorithm 4 is the constrained BO. In Fig. 4b the outputs of this stage, which correspond
Algorithm 3: CBE Path assessment

Input: \( \mathcal{U}(u) \): assessed path
\( f_{\min} \): current objective minimum

Output: \( P_{\text{valid}}, P_{\text{reward}}, f_{\min} \)

\( P_{\text{valid}} \leftarrow \text{Check: Motion Constraints} \)
\( P_{\text{valid}} \leftarrow \text{Check: Safety} \)
if \( P_{\text{valid}} \) then
  \( P_{\text{reward}} \leftarrow \text{Evaluate reward: eq. [19]} \)
else
  Path assessment(valid subset of \( P \))
end

Algorithm 4: CBE

/* Generate initial training set: */
\( N \) = Size of training set
\( \Omega \leftarrow \text{Generate training path set}(N) \)
for \( u_k \in \Omega \) do
  \( u_k \leftarrow \text{Path assessment}(u_k) \) (Algorithm 3)
  Update GP and GPCs: \( u_k \)
end
/* Constrained BO: */
for \( i = 1, 2, 3, \ldots \) do
  feasible region \( C \):
  \( C(u) = \prod_{k=1}^{K} \Pr(C_k(u) < 1 - \delta_k) \)
  Find: \( u_i \leftarrow \arg\min_{u \in C} C(u) \cdot LCB(u) \)
  \( u_i \leftarrow \text{Path assessment}(u_i) \) (Algorithm 3)
  Update GP and GPCs: \( u_i \)
end
Execute optimal path

to the paths suggested by the optimiser, are depicted in dark green. With every attempt, BO updates the GP and GPCs and becomes more confident in the model of the objective function, the constraints and the location of the global minimum. This learning process is evident from the distribution of the suggested paths. Although most are bundled around two main directions, there are some stray paths that check potentially rewarding options. Also, some paths are on the borders of the unexplored regions, suggesting the optimiser tries to learn about the motion constraints. The final output of the optimiser, the optimal path where the accumulated reward is maximised, is shown in black.

To gain additional insight into the optimisation process, Fig. 5 presents images of key CBE elements. As CBE is a high-dimensional optimisation process, cross-sections are used for their visualisation. The first key element of CBE is the surrogate GP model of the objective function, shown in the first two cross sections, \( \mu_{GP} \) and \( \sigma_{GP} \). The GP represents our belief about the learned objective function. The main benefit of using a GP, as with other Bayesian regression techniques, is the ability to obtain an inference confidence measure. The non-parametric structure gives great flexibility in expressing the model expected value and variance around observations. Previous method using BO for exploration, e.g. in (Martinez-Cantin et al. 2009),
Figure 5. Images (cross sectional) of the various components of CBE for the scenario shown in Fig. 4. Rows depict different component of BO: GP regression mean, GP standard deviation, unconstrained BO acquisition function, turn rate GPC, safety GPC and the CBE acquisition function. Columns show two orthogonal cross sections. Contours in GPC images represent valid (white) and invalid (black) thresholds. Black contours for the CBE acquisition function (last row) define valid regions for optimisation given observations. CBE will try to maximise acquisition function in valid regions, which produces a suggestion for the next observation point.
optimises using the unconstrained acquisition function which is shown inverted in Fig. 5 for clarity. Instead of only optimising over the expected value, the use of an acquisition function incorporates the model uncertainty. However, unconstrained BO is not suitable for autonomous exploration, as reward samples can only be acquired along valid trajectories. Beyond the valid region, the GP model provides only its intrinsic parameters; the model mean and maximum variance. Consequently, this breaks the internal BO feedback loop of sampling and updating, as the GP model is kept unchanged after any invalid sample.

The constrained BO framework is more suitable for autonomous exploration. The learned constraints, shown as GPCs in Fig. 3 provide the optimiser with an additional layer that incorporates invalid samples without the need to define a closed-form expression for the constraints. Using GPCs provides an efficient method to query the certainty in which the constraints are met. Furthermore, the user can easily modify the validity threshold (shown as black and white lines), to adjust the optimiser risk-reward balance. The combined acquisition function, which is shown in the last cross-section of Fig. 5 is simply the unconstrained BO acquisition function overlaid with the GPCs valid zone. Unlike unconstrained BO, integrating new invalid samples will not break the BO feedback loop. It will instead modify the GPCs. This behaviour allow the CBE to explore promising paths that are on the borders between valid and invalid.

Finally, we conclude this section with an estimate of the computational complexity of CBE. As GPs are used extensively through out this algorithm, it is not surprising that the CBE computational costs are mainly associated with GP inference complexity. Similarly to other non-parametric methods, the computational complexity depends on the size of training set, \( n \). In the CBE framework, however, two separate training sets are defined; (i) \( N \) includes the entire training set of valid and invalid points. (ii) \( m (m < N) \) includes only the valid points used by the reward GP. The computational complexity of a typical GP is \( O(n^3) \) and is due to the Cholesky decomposition of the covariance matrix [Rasmussen and Williams 2006]. GP prediction carries a lower complexity of \( O(n^2) \) arising from the solution of a triangular linear system. In CBE, on the other hand, we employ \( c \sim O(1) \) GPCs. Hence the overall complexity of the Cholesky decomposition of the various components of CBE is \( O(m^3 + cN^3) \). In addition, during optimisation, GP and GPCs model may be queried repeatedly leading to \( O(Mm^2 + cPN^2) \), where \( P \) and \( M \) are the number of queries of GPCs and GP, respectively, and \( m < P \). Given \( m < N \) the overall complexity of CBE, Cholesky decomposition and model queries, can be estimated as \( O(PN^3 + cPN^2) \). We can further simplify this expression, by noting that the typical training set contains several hundred points, as is the number of optimisation steps; \( P \sim N \). Thus, we can concisely write the overall CBE complexity as \( O(cN^3) \).

### 4.5 Incorporating Uncertainty in CBE

In the context of autonomous exploration, path planning is a decision making process aimed at improving the map fidelity. Any uncertainty, whether it is in sensor observations or in the robot pose, propagates into our belief over the map and corrupts it. While sensor uncertainty is typically a fixed limitation arising from the system configuration, the robot location uncertainty is controlled by the robot’s decisions. Reducing pose uncertainty is commonly addressed in the literature by incorporating a "loop closing" heuristic in the optimisation of the next observation point (for example [Makarenko et al. 2002, Indelman et al. 2015, Rafieisakhaei et al. 2016]). With the standard BO framework, incorporating such a heuristic requires modifications to the forward simulation reward calculation as described in the work of [Martinez-Cantin et al. 2009]. However, uncertainty in the robot pose necessitates some additional adaptations to the CBE algorithm to ensure the safety of resulting path. Therefore, we will leave the "loop closing" reward modification for future work, and discuss the required changes to CBE in the following section.

At the end of optimisation, CBE returns an optimal path. When considering only the nominal pose, the optimal path is safe and valid. However, the actual outcome of that path, and more importantly, its safety, depends on the real pose of the robot. Fig. 6 depicts an example of such a case, by plotting the same path for several starting poses, drawn from the robot’s state distribution. It is clear, that by not incorporating the pose uncertainty, the risk of collision is greatly under-estimated.

To better estimate the safety risk, we need to project the variance of the robot’s location and orientation into the safety GPC model. However, the resulting probability density function might have a non-trivial form. An efficient solution to alleviate this problem utilises an unscented transform (UT) [Julier and Uhlmann 1997]. UT employs a deterministic sampling schedule to estimate the mean and variance of the desired distribution. The sample set, termed ‘sigma points’, consists of \( 2n + 1 \) samples and weights for a \( n \)-dimensional space. Given the pose mean, \( \mu_p \) and covariance, \( \Sigma_{pp} \), the ‘sigma’ points, \( \chi_i \), are defined by the following equations:

\[
\chi_i = \begin{cases} 
\mu_p & i = 0 \\
\mu_p + (\sqrt{(n + \lambda)\Sigma_p})_i & i = 1, ..., n \\
\mu_p - (\sqrt{(n + \lambda)\Sigma_p})_i-n & i = n + 1, ..., 2n 
\end{cases}
\]

(20)
Here $\lambda = \vartheta^2(n + \nu) - n$. $\vartheta$ determines the spread of the sigma points around the mean, $\mu_p$, and is typically a small positive number (in our experiment $\vartheta = 10^{-3}$). $\nu$ is a second order term to adjust kurtosis and is usually set to zero (Wan and Van Der Merwe 2000).

For the sigma weights, we follow Wan and Van Der Merwe (2000), and define separate values for mean and covariance calculations:

$$
\begin{align*}
    w_0^{\text{mean}} &= \frac{\lambda}{n + \lambda} \\
    w_0^{\text{cov}} &= \frac{\lambda}{n + \lambda} + (1 - \psi^2 + \vartheta) \\
    w_i^{\text{cov}} &= \frac{1}{2(n + \lambda)} \\
    i &= 1, \ldots, 2\delta
\end{align*}
$$

The parameter $\vartheta$ is used to encode prior knowledge of initial distribution. Since we assume a Gaussian distribution for the pose, we take $\vartheta = 2$.

These sigma points serve as starting poses for alternative path outcomes as shown in Fig. 6. We employ the same mapping from actions to trajectory space, $\xi(u)$, but replace the implicit noiseless pose with the sigma points $\xi(u, \chi_i)$. As a result, we probe how $\xi$ changes the shape of the initial pose uncertainty and thus recover a stochastic estimate for the robot pose along the path as, $x \sim N(\rho, \Sigma_x)$:

$$
\begin{align*}
    \rho &= \sum_{i=0}^{2n} w_i^{\text{mean}} \xi(u, \chi_i) \\
    \Sigma_x &= \sum_{i=0}^{2n} w_i^{\text{cov}} (\xi(u, \chi_i) - \rho)(\xi(u, \chi_i) - \rho)^T
\end{align*}
$$

Given $\xi(u, \chi_i)$ and map, $m$, one can now stochastically reason about the safety of an action, $u$. Although straightforward, Algorithm 5 simplifies the safety estimation even further. Instead of inferring the pose distribution along path, each sigma path, originating from a specific sigma point, is validated separately. The overall validity of an action, $u$, is then determined by the worst-case scenario over all paths.

5 Results

In this section, we evaluate the performance of CBE in simulation and with a real robot.

5.1 Simulations

We divide our simulation experiments into two categories. First, the robot maps various randomly generated cluttered unstructured environments. The second experiment involves exploration of a large scale complex networks of city roads. In both cases, we simulate a ground robot equipped with a 180$^\circ$ field of view (FOV) laser scanner driving at a constant speed. The turn rate of the robot is limited, forcing the optimiser to plan within the robot’s kinematic envelope using quadratic splines. In all simulations, we assume full knowledge of the robot’s pose.

5.1.1 Unstructured Environments

Many exploration experiments involve structured man-made scenarios. A structured environment is constructed of a network of corridors, for example, underground mines and buildings. Although the unexplored regions include obstacles, the corridor-like structure pulls the robot toward an obvious general path. Unstructured scenes, with randomly positioned obstacles, break any large-scale formation, and thus lack this implicit guidance. Furthermore, these scenarios exhibit additional difficulties,
such as isolated areas with only a single access point, nontraversable narrow gaps, and long barriers dividing the world into several almost independent parts. Such an arrangement complicates the exploration process, since it introduces many more options the robot has to choose from. Fig. 7 shows examples of randomly generated worlds used to compare constrained BO exploration to other techniques.

A qualitative comparison between CBE and other map building techniques is shown in Fig. 8. The methods used for comparison follow the common exploration paradigm, where the path is determined in two separate stages:

i Selecting the next observation point.

We employ two method for the selection of the next observation point: a frontier-based method ([Yamauchi 1997] and an information-theoretic approach based the information gain utility of ([Makarenko et al. 2002]).

ii Planning a safe path to that goal point.

To emphasise the importance of the path, and not only of the end goal point, we employ two separate path planning techniques for stage two; $A^*$ planner which finds the shortest traversable path to the goal point and a fast greedy planner using the distance to the goal point as its heuristic. Both path planners enforce the robot’s safety and manouevrability limitations, by generating a path from a valid set of motion primitives.

The main advantage of CBE visible in Fig. 8, is that the number of overlapping paths is smaller when compared to the other planners. The BO planner takes a relatively short path that minimises the time the robot moves through already visited parts of the environment. CBE maximises the accumulated information gain at every decision point by avoiding previously traversed paths, which is achieved by choosing paths without explicitly defining an end goal point. In contrast to CBE, both information-theoretic and frontier based planners exhibit a clear criss-cross pattern in the executed paths, regardless of the path planner used. This suboptimal performance arises from the two stage exploration process. Choosing a goal point first and then planning a path, prevents reasoning on the potential reward along the driven path. As a result, the knowledge gained while travelling to the goal point is not considered in the decision making.

The type of path planner used has also great impact on the overall exploration performance. As expected, the greedy path planner is less effective at finding a path through the clutter, evident by the tangled paths around obstacles. This leads to longer paths with an overall lower rate of improvement. Fig. 9 provides a quantitative comparison of the rate of reduction in the map’s entropy between the various methods. The initial rate, in the first 10 seconds, is similar in all methods as the robot passes through the unexplored map. However, the rate at which the entropy decreases in the frontier and information gain based methods becomes slower, coinciding with the robot travelling through already explored regions on its way to the planner’s goal point. This outcome is independent of the path planner used, $A^*$ or greedy. The difference in performance stems from the objective of both the information-theoretic and frontier planners; to find the next global observation point. With a complex unstructured scenario, there are many potential observation points at every decision. It is clear that by visiting these points, the entire environment will be mapped. However, with global point planners there are no guarantees on the optimality of that process, as there is little reasoning about the executed path. While these techniques might put a cost or penalty on the driving distance, the gains along a path are not taken into consideration. CBE, on the other hand, plans in its local neighbourhood taking into account the benefits and risks of potential paths, rather than selecting goal points. The global component pulling the robot toward the nearest frontier only affects the decision when the local information component is negligible.


Figure 8. Comparison of simulation results for a randomly generated world (world 3). (left) CBE results, (right) simulation results with two planners for next observation point; Information-theoretic \cite{Makarenko2002} and frontier \cite{Yamauchi1997}, and two smooth safe path planners, greedy and A*. The walls and obstacles are marked with black lines. In the grid map, green is unexplored regions, blue is free space and red occupied. The executed paths are shown as dashed yellow lines. CBE maximises the accumulated information gain at every decision point by avoiding previously traversed paths. In contrast, both information-theoretic and frontier based planners exhibit a clear criss-cross pattern in the executed paths, as both methods only reason on the gains of single goal point.

Table 1. Comparison of exploration time between CBE and two planners for next observation point; Information-theoretic \cite{Makarenko2002} and frontier \cite{Yamauchi1997}, and two smooth safe path planners, greedy and A*. The walls and obstacles are marked with black lines. In the grid map, green is unexplored regions, blue is free space and red occupied. The executed paths are shown as dashed yellow lines. CBE maximises the accumulated information gain at every decision point by avoiding previously traversed paths. In contrast, both information-theoretic and frontier based planners exhibit a clear criss-cross pattern in the executed paths, as both methods only reason on the gains of single goal point.

| Observation point selection | CBE | Exploration Time [s] | Information-theoretic | Frontier |
|-----------------------------|-----|----------------------|-----------------------|----------|
| Path Planner                |     |                      | Greedy                | A*       | Greedy | A*       |
| World 1                     | 63.8 (46%) | 136.1 (97%) | 68.9 (49%) | 139.9 (100%) | 85.1 (61%) |
| World 2                     | 86.2 (64%) | 134.5 (100%) | 82.7 (61%) | 87.2 (65%) | 88.5 (66%) |
| World 3                     | 86.5 (78%) | 108.1 (97%) | 98.4 (88%) | 111.6 (100%) | 99.9 (89%) |
| World 4                     | 64.7 (58%) | 106.0 (92%) | 64.5 (56%) | 115.6 (100%) | 98.1 (85%) |
| World 5                     | 79.6 (89%) | 84.4 (94%) | 89.8 (100%) | 73.5 (82%) | 78.3 (87%) |
| World 6                     | 79.4 (17%) | 454.6 (100%) | 76.9 (17%) | 120.2 (26%) | 80.2 (18%) |
| World 7                     | 62.6 (63%) | 87.9 (89%) | 65.3 (66%) | 99.0 (100%) | 86.6 (87%) |
| World 8                     | 77.1 (52%) | 112.4 (76%) | 88.7 (60%) | 147.4 (100%) | 90.7 (62%) |
| World 9                     | 71.7 (54%) | 129.0 (97%) | 115.0 (87%) | 132.4(100%) | 102.3 (77%) |
| World 10                    | 76.6 (66%) | 115.4(99%) | 98.0 (85%) | 111.0 (96%) | 95.1 (82%) |

Table 1 provides a quantitative comparison between the exploration techniques on several randomly-generated worlds shown in Fig. 7. As expected, A* is a noticeably better path planner than the greedy planner, as it guarantees
Figure 9. Quantitative comparison of the reduction in map entropy (world 3) between exploration methods presented in Fig. 8. The initial rate of entropy reduction is similar in all methods. However, the rate slows in both the information-theoretic and frontier methods, as the robot travels through already explored region in route to its next goal point. CBE, avoids previously traversed areas of the map, leading to a faster reduction in entropy.

As we assume the robot pose is fully known in these simulations, repeatability was tested by changing the initial pose. Fig. 10 and Table 2 present a comparison between CBE and frontier/A* in world 3 (see in Figs. 7 and 8) for various starting poses. Once again, we can see that CBE outperforms the competing method in most cases.

5.1.2 Structured Environments

These experiments test CBE performance in a structured environment scenario. Part of the roads and paths network of Venice and Jerusalem old city were extracted from Google Maps. These complex networks of corridor-like patterns serve as the ground truth in this large-scale exploration experiment. In such a structured system, there is no clear advantage for the constrained BO method. The corridor structure forms an obvious path, which limits the local significance of path selection. As there are little differences in rewards along the paths, the end goal becomes the most important property of a path. Hence, a CBE planner would be potentially ineffective. By comparison, the frontier based-A* approach seems to be the
Table 2. Repeatability - quantitative comparison of exploration paths originating from various starting poses as shown in Fig. 10. Comparison is between CBE and frontier (Yamauchi 1997) exploration methods.

| #  | Exploration Time [s] | Diff [%] |
|----|----------------------|----------|
|    | CBE                  | Frontier |          |
| 1  | 128.9                | 107.3    | -16.8    |
| 2  | 89.8                 | 111.9    | 24.5     |
| 3  | 68.7                 | 104.6    | 52.2     |
| 4  | 96.9                 | 103.9    | 7.3      |
| 5  | 72.7                 | 97.4     | 34.1     |
| 6  | 61.1                 | 78.7     | 28.8     |
| 7  | 61.8                 | 98.6     | 59.7     |
| 8  | 88.4                 | 91.4     | 3.4      |
| 9  | 99.5                 | 98.5     | -1.0     |
| 10 | 70.0                 | 76.9     | 9.8      |

Average 83.8 96.9 20.2
std. dev. 21.2 11.6 24.1

most sensible method for such a problem, as it moves the robot on the shortest path to the edge of the known space.

Fig. 11 depicts the executed path of a robot exploring the surroundings of the Piazza San Marco, Venice, while Fig. 12 shows exploration around the Church of The Holy Sepulchre, Jerusalem. In both cases, the road network is complex, creating many possibilities for autonomous actions. From this qualitative comparison, one can see that all techniques cover almost the entire mission area (blue square) with no isolated pockets of unexplored regions. However, a closer examination of the executed paths reveals a significant difference. As the map is a-priori unknown, it is reasonable that the robot will have to move occasionally through already mapped roads. However, the paths generated by the information-theoretic heuristics revisits known roads much more than the other planners. Although the search heuristics includes a distance penalty, it is not general enough to be effective in all scenarios. Once again, the reason behind such a sub-optimal performance lies in the basic properties of the global point planners i.e. separating the solution for the goal point from the subsequent path generation. We should note that in these scenarios the frontier planner is not as affected. Choosing the closest frontier as the planner’s goal point keeps path planning in the robot’s local neighbourhood. However, such an arbitrary goal point selection can not always guarantee an optimal path.

A quantitative comparison for the Venice and Jerusalem exploration simulations are shown in Figs. 13 and 14 respectively. Most importantly, even in unfavourable conditions, CBE achieves performance as good as frontier. A careful inspection of the results reveals additional insights.
Figure 12. Comparison of simulation results of map building in Jerusalem old city. Left to right; CBE, frontier and information-theoretic. Traversable roads are extracted from Google Maps. The blue background marks the allowed region. Explored area are shown without the blue background. The path the robot executed is in yellow, and the last position of the robot is marked with a red asterisk. As expected, all methods explored almost the entire mission area. The paths generated by the information-theoretic method revisits explored region of the map much more than the other methods, although a distance penalty is incorporated in its reward heuristic. CBE and frontier present similar path structure as both plan mostly in the robot's close neighbourhood.

Figure 13. Venice - Comparison of reduction in map entropy between exploration methods presented in Fig. 11. The overall time to cover the mission area is similar with both CBE and frontier. Both methods outperform the information-theoretic method as the number of paths crossing already explored regions of the map is lower.

Figure 14. Jerusalem - Comparison of reduction in map entropy between exploration methods presented in Fig. 12. The rate of entropy reduction using CBE and Frontier is similar and outperforms the information-theoretic method, which corresponds to lower instances of crossing already explored regions of the map.

Similarly to the unstructured environment simulations, the initial rate in which the map entropy drops is similar in all techniques. As the robot moves through the map, the number of possible actions increases leading to differences in performance. In Venice, frontier is less effective at first, while in Jerusalem the information-theoretic solution is less effective. Yet, in both experiments the CBE algorithm kept a consistent performance comparable or better than the other leading method regardless which one it is. These results affirm the conclusion from our previous experiment that constrained BO is a robust exploration method.

The essence of the CBE method lies in its reasoning about the usefulness and safety of the entire path taken. The benefits of reasoning on the overall accumulated reward are more distinct when compared to the information-theoretic single goal point exploration technique (red line). The qualitative results shown in Figs. 11 and 12 expose the inefficiencies in the single goal point methods. The slower exploration rate in this method corresponds to the revisiting of already explored regions of the map whilst moving to the next goal point. By reasoning on the path utility instead of the end goal point, CBE avoids selecting paths that provide
little information. Furthermore, the global component in the constrained BO reward function is found to be very effective in pulling robot away from dead-ends. Yet, a more expressive path option is more desirable in such a case, since it will allow longer term planning.

5.2 Real Environments

Simulations show the effectiveness of CBE for autonomous exploration. However, to assess performance with partially observed poses, CBE was evaluated with a real robot mapping a cluttered office environment. We used our in-house robot, the Wombot (see Fig. 15), equipped with an i7-4500U 1.8GHz dual-core on-board PC and an Hokuyo UTM-30lx laser range finder.

![The Wombot - a mobile robot equipped with a Hokuyo UTM-30lx laser range finder.](image)

We use Robot Operating System (ROS) (Quigley et al. 2009) to manage the communication between the various components of the robot i.e. sensors, actuators etc., and software modules. For mapping and localisation, we utilise externally provided ROS package, gMapping (Grisetti et al. 2007). It is worth noting that the aim of these experiments is to assess the performance in the presence of pose uncertainty. Although CBE can encode loop-closing heuristic in its objective function, we only used the objective function defined in Algorithm 5.

Fig. 16 shows the maps and paths taken at different time stamps for both CBE and frontier. Similarly to experiments in simulations, the frontier planner places a goal point at the closest frontier. Fig. 16 shows that both methods cover the entire space. However, while both stay clear of obstacles, the path taken by the frontier planner is less efficient. The occluded space behind the various obstacles forms frontiers that are then visited by the robot. As the robot visits these goal points, the overall path length and exploration time increase. CBE, on the other hand, considers the utility of the entire path as opposed to only considering the utility of the goal point. Furthermore, assessing safety using sigma paths that considers the effect of the uncertain pose with robot motion proved to be successful.

Fig. 17 provides a quantitative comparison between the two methods. In the beginning, both methods perform similarly. After about 50 seconds the two methods start to diverge as the frontier planner pulls the robot towards a goal point behind an obstacle. As the exploration continues, the performance gap between the two methods increase, mainly due to a non-optimal goal point selection by the frontier planner. The final map, and the reduction in overall entropy is the same in both methods, although it took frontier roughly 30 seconds longer to do so.

The advantage of CBE comes with a computational cost as shown in Table 3. Finding a frontier and planning a safe path from the robot pose to a specific goal point depends of the size of the map and the distance to the goal point. Typically in our experiments, planning took less than 1 second. However, the search for a safe path took longer when there was no safe path to the selected frontier. In such a case, a safe path to a different goal point was calculated, but only after the first search had exhausted its time budget. CBE is more stable despite the higher computational cost. Apart from the first planning instance, which includes a hyperparameters training stage, all the subsequent instances took slightly less than a minute. As shown in section 4.3, updating and querying the GPCs is the main the culprit. Choosing a different constraint representation has the potential to dramatically improve CBE’s speed.

6 Conclusions

6.1 Conclusions

This paper proposes a new strategy for safe autonomous exploration. Its novelty lies in the holistic probabilistic approach to robotic exploration. Specifically the paper presents the following contributions:

(i) Formulation of autonomous exploration as a Bayesian optimisation problem in constrained continuous space, where the path is evaluated by its accumulative reward and not only by the reward of its goal point. Traditional exploration methods consist of a two-step solution. First, a collection of goal points (typically one) is defined by a set of heuristics, followed by a path planning step. As a result, the expected usefulness of the resolved path is based solely

| Table 3. Planning and execution time comparison |
|-----------------------------------------------|
|                         | BO       | Frontier |
|-------------------------|----------|----------|
| 1st plan [s]            | 164      | -        |
| Average Plan [s]        | 58.4     | 9.8      |
| Execution [s]           | 192      | 236      |
Figure 16. Map building comparison in a real environment. Each image represents the available map at a specific time stamp shown on the left. The frontier method produces a map less efficiently as it chooses unnecessary long paths due to frontiers forming in occluded space behind the various obstacles.
on the utility function of the end point and does not consider any potential gains along the way. Our new strategy, on the other hand, does not set goal points. Rather, it optimises the path selection by learning the properties of the objective function and any associated constraints. Consequently, the full potential of the robot trajectory can be exploited and not only that of the end point.

(ii) Constrained Bayesian Exploration as a holistic approach to safe exploration. This method directs the optimisation process in the presence of unknown constraints and risks. Utilising Bayesian inference, the optimiser learns the models of the rewards and constraints. These models are then used to generate a coherent objective function that incorporates gains, costs and risks of any path, allowing efficient identification of potential optimal solutions that satisfy the constraints with high confidence. Unlike traditional exploration techniques, constraints are an integral part of the search mechanism. As the actual model of the constrains is learned online, its incorporation in the optimisation utility is straightforward. Hence, CBE allows a relatively simple and smooth application of other limitations such as energy and time budget. In addition, the probabilistic representation of the constraints can allow the user to balance risk and gain in the process of exploration.

To test the efficiency of our method, we compared its performance with other exploration techniques. The results show that the performance of each of the other exploration techniques depends on the layout of the environment. By reasoning on the usefulness of the entire path instead of only its goal point, CBE exhibits a robust and consistent performance in all tests. Even in unfavourable conditions of structured environments, CBE performs better than or as good as the leading method.

The use of sigma paths to incorporate localisation uncertainty proved successful. A robot travelling through cluttered office space managed to avoid obstacles while still optimising the cost function.

6.2 Future Work

The main disadvantage of CBE is the computational cost. In the algorithm GPCs are the main computational bottleneck as the cost of updating and querying GPCs is cubic in the number of training points. A promising approach to alleviate this restriction is to utilise stochastic variational inference for GPC (Hensman et al. 2015), which has a computational cost independent of the number of data points.

Another avenue for future work is in extension of the CBE framework to a more flexible family of trajectories. The use of a predefined family of trajectories, such as quadratic or cubic splines, limits the decision space of the robot. This problem is more pronounced when path planning is constrained, for example near obstacles, as the optimisation space is confined. Examples of more expressive path generation already exists in the literature (Yang et al. 2013; Charrow et al. 2015). However, these methods do not consider the overall reward along the path, or only locally optimise the path selection. Combining CBE with an RKHS motion planning method, such as the method presented by Marinho et al. (2016), may allow for global optimisation of highly expressive paths.

References

Basilico N and Amigoni F (2011) Exploration Strategies based on Multi-Criteria Decision Making for an Autonomous Mobile Robot. Int. Conf. on Autonomous Agents and Multiagent Systems: 1–6.

Bourgault F, Makarenko A, Williams S, Grocholsky B and Durrant-Whyte H (2002) Information based adaptive robotic exploration. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), volume 1. pp. 540–545.

Brochu E, Cora VM and De Freitas N (2010) A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. preprint arXiv:1012.2599.

Charrow B, Kahn G, Patil S, Liu S, Goldberg K, Abbeel P, Michael N and Kumar V (2015) Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping. In: Proc. Robotics: Science and Systems (RSS).

Dornhege C and Kleiner A (2013) A frontier-void-based approach for autonomous exploration in 3D. Advanced Robotics 27(6): 459–468.
Elfes A (1996) Robot navigation: Integrating perception, environmental constraints and task execution within a probabilistic framework. In: Reasoning with Uncertainty in Robotics: International Workshop (RUR ’95). Amsterdam, The Netherlands, pp. 91–130.

Gelbart MA, Snoek J and Adams RP (2014) Bayesian Optimization with Unknown Constraints. In: Uncertainty in Artificial Intelligence (UAI).

González-Baños HH and Latombe J-C (2002) Navigation Strategies for Exploring Indoor Environments. The International Journal of Robotics Research 21(10-11): 829–848.

Gramacy RB and Lee HKH (2011) Optimization Under Unknown Constraints. In: Bayesian Statistics 9. Oxford University Press, pp. 229–256.

Grissett G, Stachniss C and Burgard W (2007) Improved techniques for grid mapping with Rao-Blackwellized particle filters. IEEE Transactions on Robotics 23(1): 34–46.

Hensman J, Matthews A and Ghahramani Z (2015) Scalable Variational Gaussian Process Classification. In: AISTATS.

Indelman V, Carlone L and Dellaert F (2015) Planning in the Continuous Domain : a Generalized Belief Space Approach for Autonomous Navigation in Unknown Environments. The International Journal of Robotics Research 34(7): 1–56.

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

Julian BJ, Karaman S and Rus D (2014) On mutual information-based control of range sensing robots for mapping applications. The International Journal of Robotics Research 33(10): 1375–1392.

Julier SJ and Uhlmann JK (1997) New extension of the Kalman filter to nonlinear systems. In: AeroSense’97. pp. 182–193.

Kollar T and Roy N (2008) Trajectory Optimization using Equation-based Exploration Algorithm for Autonomous Indoor Environments. The International Journal of Robotics Research 27(2): 175–196.

Lauri M and Ritala R (2015) Planning for robotic exploration based on forward simulation. preprint arXiv:1502.02474.

Mahonen AA, Williams SB, Bourgault F and Durrant-Whyte HF (2002) An experiment in integrated exploration. In: IEEE/RSJ Int. Conf. on Intelligent Robots and System (IROS), volume 1. pp. 534–539.

Marchant R and Ramos F (2014) Bayesian Optimisation for Informative Continuous Path Planning. In: Proc. IEEE Int. Conf. Robotics and Automation (ICRA). pp. 6136–6143.

Marchant R, Ramos F and Sanner S (2014) Sequential Bayesian optimisation for spatial-temporal monitoring. In: Uncertainty in Artificial Intelligence (UAI).

Marinho Z, Boots B, Dragan A, Byravan A, Gordon GJ and Srinivasa S (2016) Functional Gradient Motion Planning in Reproducing Kernel Hilbert Spaces. In: Proc. Robotics: Science and Systems (RSS).

Martínez-Cantín R, de Freitas N, Brochu E, Castellanos J and Doucet A (2009) A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. Autonomous Robots 27(2): 93–103.

Martínez-Cantín R, de Freitas N, Doucet A and Castellanos J (2007) Active Policy Learning for Robot Planning and Exploration under Uncertainty. In: Proc. Robotics: Science and Systems (RSS).

O’Callaghan ST and Ramos FT (2012) Gaussian process occupancy maps. The International Journal of Robotics Research 31(1): 42–62.

Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, Wheeler R and Ng AY (2009) ROS an open-source Robot Operating System. In: ICRA workshop on open source software, volume 3. p. 5.

Rafieisakhaei M, Tamjidi A, Chakravorty S and Kumar PR (2016) Feedback motion planning under non-Gaussian uncertainty and non-convex state constraints. In: Proc. IEEE Int. Conf. Robotics and Automation (ICRA). pp. 4238–4244.

Ramos F and Ott L (2015) Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent. In: Proc. Robotics: Science and Systems (RSS).

Rasmussen CE and Williams CKI (2006) Gaussian Processes for Machine Learning. MIT Press.

Shade R and Newman P (2011) Choosing where to go: Complete 3D exploration with stereo. In: Proc. IEEE Int. Conf. Robotics and Automation (ICRA). pp. 2806–2811.

Shen S, Michael N and Kumar V (2012) Stochastic differential equation-based exploration algorithm for autonomous indoor 3D exploration with a micro-aerial vehicle. The International Journal of Robotics Research 31(12): 1431–1444.

Stachniss C (2009) Robotic Mapping and Exploration. Springer.

Stachniss C, Grisetti G and Burgard W (2005) Information Gain-based Exploration Using Rao-Blackwellized Particle Filters. In: Proc. Robotics: Science and Systems (RSS).

Thrun S, Burgard W and Fox D (2005) Probabilistic Robotics. MIT Press.

Tovar B, Muñoz-Gómez L, Murrieta-Cid R, Alencastre-Miranda M, Monroy R and Hutchinson S (2006) Planning exploration strategies for simultaneous localization and mapping. Robotics and Autonomous Systems 54(4): 314–331.

Vallvé J and Andrade-Cetto J (2015) Potential information fields for mobile robot exploration. Robotics and Autonomous Systems 76: 68–79.

Wan E and Van Der Merwe R (2000) The unscented Kalman filter for nonlinear estimation. In: Proc. IEEE Symposium 2000 (AS-SPCC). pp. 153–158.

Whaite P and Ferrie FP (1997) Autonomous exploration: driven by uncertainty. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(3): 193–205.
Yamauchi B (1997) A frontier-based approach for autonomous exploration. In: IEEE Int. Symp. Computational Intelligence in Robotics and Automation. pp. 146–151.

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.