Planning in the Continuous Domain: a Generalized Belief Space Approach for Autonomous Navigation in Unknown Environments

Vadim Indelman*, Luca Carlone†, and Frank Dellaert†

Abstract

We investigate the problem of planning under uncertainty, with application to mobile robotics. We propose a probabilistic framework in which the robot bases its decisions on the *generalized belief*, which is a probabilistic description of its own state and of external variables of interest. The approach naturally leads to a dual-layer architecture: an *inner estimation layer*, which performs inference to predict the outcome of possible decisions, and an *outer decisional layer* which is in charge of deciding the best action to undertake. Decision making is entrusted to a Model Predictive Control (MPC) scheme. The formulation is valid for general cost functions and does not discretize the state or control space, enabling planning in continuous domain. Moreover, it allows to relax the assumption of *maximum likelihood observations*: predicted measurements are treated as random variables, and binary random variables are used to model the event that a measurement is actually taken by the robot. We successfully apply our approach to the problem of uncertainty-constrained exploration, in which the robot has to perform tasks in an unknown environment, while maintaining localization uncertainty within given bounds. We present an extensive numerical analysis of the proposed approach and compare it against related work. In practice, our planning approach produces smooth and natural trajectories and is able to impose soft upper bounds on the uncertainty. Finally, we exploit the results of this analysis to identify current limitations and show that the proposed framework can accommodate several desirable extensions.

1 Introduction

Autonomous navigation in complex unknown scenarios involves a deep intertwining of estimation and planning capabilities. A mobile robot is required to perform *inference* from sensor measurements, in order to build a model of the

* Faculty of Aerospace Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel.

†Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA 30332, USA.

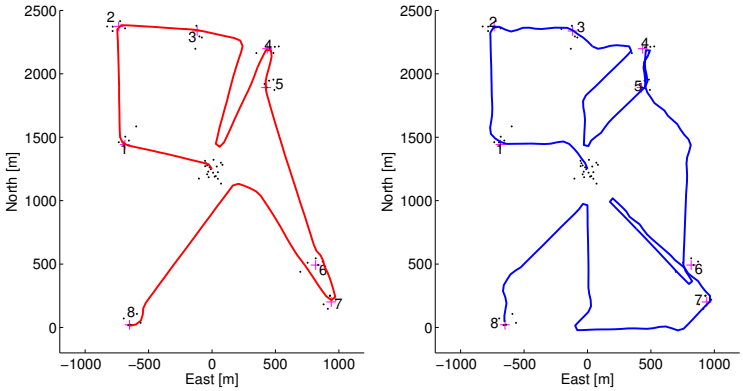
surrounding environment and to estimate variables of interest. Moreover, it has to *plan* actions, to accomplish given goals. If the environment in which the robot operates is unknown or uncertain, robot decisions has to rely on the estimates (the *world model*) coming from the inference process. This problem is an instance of *a partially observable Markov decision process* (POMDP), which describes a decisional process in Markovian systems, in which the state is not directly observable. While POMDP are intractable in general (Kaelbling et al., 1998), it is of practical interest to devise problem-specific solutions or approximations (e.g., *locally* optimal plans) that trade-off optimality for computational efficiency. Efficient and reliable planning under uncertainty is crucial in many application endeavors in which the robot operates in full or partial autonomy, ranging from autonomous exploration, monitoring and surveillance, to robotic surgery.

The estimation problem arising in robot navigation is now well understood. State-of-the-art techniques for localization and mapping (*SLAM*) allow fast solution of medium-large scenarios (Kaess et al., 2012; Konolige et al., 2010; Kümmerle et al., 2011; Kaess et al., 2008), using efficient nonlinear optimization techniques, that exploit the structure of the underlying problem. These techniques are able to manage a large number of heterogeneous measurements, and compute a posterior on robot and world state (the *belief*), that is usually parametrized in its first and second order moments.

On the other hand, planning under uncertainty is still attracting considerable attention from the robotic community, as few approaches are able to deal with the complexity and time constraints of real-world problems. The planning problem consists in establishing a map between the current belief and the action space, such that the robot can autonomously determine a suitable action (e.g., a motion command), depending on its posterior (e.g., estimate of current robot pose and landmarks positions). The complexity stems from the fact that robot observations and dynamics are stochastic: after applying a motion command, the actual robot position differs from the predicted one, due to actuation noise. Even more important, the robot does not know, a-priori, which measurements will be acquired and what the (future) sensor readings will be. A further source of complexity is that in problems such as *active sensing* and *active SLAM*, the motion strategy directly influences the amount and the quality of the acquired measurements, making planning and estimation even more coupled.

While sometimes one can simply neglect the probabilistic aspects of the problem, e.g., assuming that the current state estimate coincides with the true state, several applications require dealing with the uncertainty. For instance, in presence of large uncertainty, using the current estimate in place of the true state can lead to catastrophic failures of the system; moreover, the policy computed by the robot may have the explicit goal of minimizing some uncertainty metric (e.g., active localization, informative path planning, active SLAM).

Overview of related work: Recent literature on planning is trying to go beyond three limitations that are common in related work: *discretization*, the *maximum likelihood* assumption, and the availability *of prior knowledge of the scenario*. While we detail related work in Section 2, we anticipate an overview



(a) Planning in continuous domain (b) Planning in discrete domain

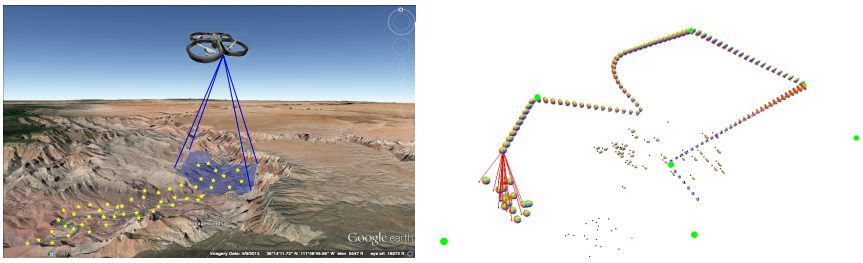
Figure 1: The robot starts from the center of the scenario, and has to visit a sequence of goals, enumerated from 1 to 8, labeled with a red +. The robot can also observe landmarks in the environment (black dots). (a) Our planner works in a continuous domain and produces the red trajectory: the robot visits goals 1-3; then, before moving to goal 4, performs a loop closure. Another loop closure is performed when moving from goal 7 to 8. (b) Planning in discrete domain produces a non-smooth and less natural trajectory (blue).

of these limitations.

Approaches relying on *discretization* approximate the state space or the space of possible control as finite sets. This approximation is usually beneficial from a computational standpoint: instead of *computing* a control action, discrete approaches *select* the control action within a finite set of *candidate* control actions. In general, this approximation sacrifices optimality, as the optimal plan may not belong to the set of candidates. The suboptimal plans, produced by discrete techniques, are usually less natural as discretization introduces artificial perturbations on the planned trajectories (Fig. 1); moreover, the performance of the approach heavily depends on the granularity of the discretization.

The second limitation is related to the assumption that the robot acquires *maximum likelihood observations*: since future observations are not given at planning time, the robot assumes that it will acquire the most likely measurements given the predicted belief. This assumption constitutes an approximation for two reasons: the actual sensor measurements will be clearly different from the expected ones; moreover, the robot does not know a-priori if a measurement will be actually acquired (e.g., in active SLAM the robot may or may not observe a landmark, depending on its position). The former issue has been recently recognized and tackled in (Van Den Berg et al., 2012).

The third limitation is connected to the assumption that some *prior knowledge of the environment* is available and the belief typically represents only robot position. While this assumption leads to effective approaches (Van Den Berg



(a) (b)

Figure 2: (a) A UAV flies in an unknown region and observes natural landmarks (yellow $\star$). (b) From sensor observations, the UAV can estimate the posterior over its own trajectory and landmarks position. The figure shows confidence ellipsoids (for robot trajectory and landmarks estimates) and current landmark observations (red lines connecting current robot position to each landmark). An accompanying video in Multimedia Extension 1 (Appendix A) shows the entire scenario considered in this figure.

et al., 2012; Prentice and Roy, 2009), in many interesting application scenarios the robot has to operate in unknown or uncertain regions. These scenarios include aerial navigation in GPS-denied environments (a simplified illustration is shown in Fig. 2a) and indoor navigation.

Paper contribution: In this work we address these three limitations. We assume the robot operates in an unknown or uncertain scenario and model this source of uncertainty as part of the belief space. Thus, contrarily to prior work on belief-space planning that typically assumes the world model is given, our extended belief space, which we call the *generalized belief space* (GBS), represents a joint probability distribution over the robot state and the state of external variables of interest, such as position of 3D landmarks in the environment (Fig. 2b). We remark that the generalized belief space is nothing more than a standard belief space over a larger state (Hauskrecht, 2000), and we use this terminology to make explicit the fact that our state space also includes the world state. Planning in GBS, similarly to planning in belief space (e.g. (Platt et al., 2010; Erez and Smart, 2010; Van Den Berg et al., 2012)), is done in a continuous domain. Moreover, we avoid the assumptions of *maximum likelihood observations,* and treat future measurements as random variables, as suggested in (Van Den Berg et al., 2012). Going beyond (Van Den Berg et al., 2012), we further model the uncertainty in the fact that a future measurement may be acquired or not; the event of acquiring a future observation is modeled as a binary random variable. Our approach is based on a Model Predictive Control (MPC) scheme: at each time step the robot plans a suitable motion strategy over a time horizon. The planner uses a dual-layer architecture: an *inner estimation layer*, which performs inference to predict the outcome of possible decisions, and an *outer decisional layer* which is in charge of deciding the best action to undertake. The formulation is valid for general cost functions. We also develop

a formulation for a specific objective function that allows to tackle the problem of uncertainty-constrained exploration, which is of practical interest in mobile robotics. We present an extensive numerical analysis of the proposed approach and use the results of the analysis to underline intrinsic limitation of our technique and of related work. In practice, our planning approach produces smooth and natural trajectories, and is shown to be very effective in a large variety of problem instances.

The present paper is an extension of the work presented in (Indelman et al., 2013) and (Indelman et al., 2014). As a further contribution, in this manuscript we present an extensive experimental evaluation (Section 7), with the aim of testing the proposed approach in a large variety of challenging scenarios. This paper also includes proofs (e.g, the derivation in Appendix B), and examples (e.g., Section 8.1) that were omitted in the conference versions for space reasons. Finally, here we try to give the reader some more insight on the problem, discussing theoretical limitations and practical remarks regarding the proposed approach.

Paper structure: The paper is organized as follows. Section 2 reviews related work. Section 3 introduces the concept of *generalized belief space*, and gives a formal statement of the problem. Section 4 presents our planning approach, which is applicable to general cost functions. Section 5 clarifies the derivation by tailoring the approach to a specific cost function. Section 6 provides implementation details, including the pseudo-code of our implementation. Section 7 presents experimental results, evaluating the proposed approach and comparing it against related work. Section 8 discusses possible extensions of the proposed approach. Conclusions are drawn in Section 9.

2 Related Work

The robotics literature on planning under uncertainty offers many heterogeneous contributions. A recurrent idea is to restrict the state space or the control space to few possible values. While *discretization* prevents obtaining an optimal solution, it allows to frame the "computation" of a plan into a "selection" of the best plan among few candidates. This usually implies a computational advantage. A consistent research effort has been devoted to design suitable metrics to quantify the quality of a candidate plan. The problem itself is not trivial as the metric depends on the representation used for the environment and on the inference engine. Examples of this effort are the work of Stachniss et al. (Stachniss et al., 2004, 2005), Blanco et al. (Blanco et al., 2008), and Du et al. (Du et al., 2011), in which particle filters are used as estimation engine. The basic idea, in this case, is to identify potential targets in the scenario (e.g., boundaries of unexplored regions, or loop closure opportunities), and to compute a path to each of those targets; the planner "simulates" the posterior along such paths, and selects the best path as the one that maximizes

a given objective function. Martinez-Cantin et al. (Martinez-Cantin et al., 2007) and Bryson and Sukkarieh (Bryson and Sukkarieh, 2008) investigate planning techniques in conjunction with the use of EKF for estimation. Martinez-Cantin et al. (Martinez-Cantin et al., 2009) propose a Bayesian optimization method that allows reducing the number of posterior evaluations. Huang et al. (Huang et al., 2005) propose a model predictive control (MPC) strategy, associated with EKF-SLAM. Leung et al. (Leung et al., 2006) propose an approach in which the MPC strategy is associated with a heuristic based on global attractors. Sim and Roy (Sim and Roy, 2005) propose A-optimal strategies for solving the active SLAM problem. Carrillo et al. (Carrillo et al., 2012) provide an analysis of the uncertainty metrics used in EKF-based planning. Other examples are (Kim and Eustice, 2013; Valencia et al., 2011, 2013) in which the belief is assumed to be a Gaussian over current and past poses of the robot.

While the approaches mentioned above are applied to mobile robot navigation (the corresponding problem is referred to as *active Simultaneous Localization and Mapping*), similar strategies can be found in the manipulation and computer vision domains (e.g., *next best view* problem (Potthast and Sukhatme, 2011)). For instance, Kaelbling et al. (Kaelbling and Lozano-Pérez, 2011, 2012, 2013) propose an approach based on hierarchical goal regression for mobile manipulation. A related problem is also the so-called *informative path planning* (although in these problems the estimation aspects are often neglected). A greedy strategy for informative path planning is proposed by Singh et al. (Singh et al., 2009), while a branch and bound approach is proposed by Binney et al. in (Binney and Sukhatme, 2012). Hollinger et al. (Hollinger and Sukhatme, 2013) propose more efficient algorithms, based on rapidly-exploring random trees and probabilistic roadmaps. These approaches usually assume that the robot moves in a partially known environment; a remarkable property is that the solutions, produced by those techniques, approach optimality when increasing the runtime (which is exponential in the size of the problem).

Most of the algorithms discussed so far assume a specific cost function and the optimization is performed over a *discrete* set of candidate controls or states (e.g., robot position belongs to a uniformly-spaced grid). However, as mentioned earlier, reasoning on a discrete space often leads to suboptimal paths (Fig. 1). For this reason, recent efforts of the research community are pushing towards the use of continuous-domain models in which commands belong to a continuous set. Continuous models appear as more natural representations for real problems, in which robot states (e.g., poses) and commands (e.g., steering angles) are not constrained to few discrete values. In (Bai et al., 2013), Bai et al. avoid discretization by using Monte Carlo sampling to update an initial policy. Platt et al. (Platt et al., 2010) apply linear quadratic regulation (LQR) to compute locally optimal policies. Erez and Smart (Erez and Smart, 2010) also calculate locally optimal policies, considering continuous state, action and observation spaces and approximating the belief space with a mixture of Gaussians. Kontitsis et al. (Kontitsis et al., 2013) propose a sampling-based approach to solve a constrained optimization problem in which constraints correspond to state dynamics, while the objective function rewards uncertainty reduction and goal

attainment.

While continuous-domain planning techniques have already produced excellent results, they still rely on two basic assumptions: (i) future observations are assumed to reflect current robot belief (*maximum likelihood observations*), and (ii) the environment in which the agent moves is partially or completely known. In the work (Van Den Berg et al., 2012) Van den Berg et al. deal with the former issue and propose a general planning strategy in which maximum likelihood assumption is relaxed. However, (Van Den Berg et al., 2012) still assumes prior knowledge of the environment, and robot belief only encodes uncertainty in robot state (e.g., robot position).

In the present work, we extend related literature in several directions. First, we avoid discretization, by working on continuous control actions. Second, we relax the assumption of prior knowledge of the environment and let the robot to operate in a completely unknown or uncertain scenario. Third, we borrow the derivation of (Van Den Berg et al., 2012) to relax the assumption of maximum likelihood observation, and adapt it to the case in which the robot keeps a joint belief over its own state and the state of the observed environment. As a further extension to (Van Den Berg et al., 2012), we model the fact that, when the robot operates in an unknown or uncertain environment, it may fail to acquire some of the expected measurements (e.g., a landmark is further than expected and is not observed, as it falls outside the sensing range).

3 Preliminaries and Problem Formulation

3.1 Notation and Preliminaries

Let x_i and $\mathcal{W}_i$ denote the *robot state* and the *world state* at time t_i. For instance, in mobile robots navigation, x_i may describe robot pose at time t_i and $\mathcal{W}_i$ may describe the positions of landmarks in the environment observed by the robot by time t_i. In a manipulation problem, instead, x_i may represent the pose of the end effector of the manipulator, and $\mathcal{W}_i$ may describe the pose of an object to be grasped. Let Z_i denote the available observations at time t_i and u_i the control action applied at time t_i. We define the joint state at time t_k as $X_k \doteq \{x_0, \ldots, x_k, \mathcal{W}_k\}$, and write the probability distribution function (pdf) over the joint state as:

$$p\left(X_k | \mathcal{Z}_k, \mathcal{U}_{k-1}\right), \tag{1}$$

where $\mathcal{Z}_k \doteq \{Z_0, \ldots, Z_k\}$ represents all the available observations until time t_k, and $\mathcal{U}_{k-1} \doteq \{u_0, \cdots, u_{k-1}\}$ denotes all past controls. We remark that the above definition of the state X_k assumes a static environment, as it only includes $\mathcal{W}_k$, the environment state observed by the current time t_k. This is a fairly standard assumption in SLAM literature. Although conceptually, dynamically changing environments can also be supported within a smoothing formulation (see, e.g. (Huang et al., 2014)), in practice it would lead to a huge state space.

The probabilistic motion model given control u_i and robot state x_i is

$$p\left(x_{i+1} | x_i, u_i\right). \tag{2}$$

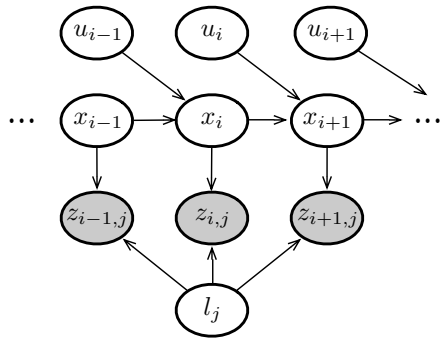


Figure 3: Graphical model describing motion (2) and observation (3) models. For simplicity, the figure shows observations of only a single landmark l_j, indicating it is observed at time instances t_{i-1}, t_i and t_{i+1}.

At a given time step t_i, the robot may acquire multiple observations, and for this reason we write $Z_i = \{z_{i,1}, \ldots, z_{i,n_i}\}$, where $z_{i,j}$ is a single observation and n_i is the number of such observations. For instance, the robot may observe different landmarks at a given time step, and $z_{i,j}$ could represent observation of the jth landmark. We consider a *general* observation model for each $z_{i,j} \in Z_i$, and denote by a superscript 'o' the involved subset of joint states $X^o_{i,j} \subseteq X_i$:

$$p\left(z_{i,j}|X^o_{i,j}\right). \tag{3}$$

For instance, $X^o_{i,j}$ may include current robot pose and the position of the jth landmark. The motion (2) and observation (3) models are shown in graphical form in Figure 3, which for simplicity includes only a single landmark l_j. An even simpler observation model, commonly used in motion planning, e.g., (Van Den Berg et al., 2012), involves only the current robot state x_i at each time t_i and is a particular case of the general model (3).

The joint pdf (1) at the current time t_k can be written according to the motion (2) and observation (3) models as (see graphical model in Figure 3):

$$p\left(X_k|\mathcal{Z}_k, \mathcal{U}_{k-1}\right) = priors \cdot \prod_{i=1}^{k} \left[p\left(x_i|x_{i-1}, u_{i-1}\right) \prod_{j=1}^{n_i} p\left(z_{i,j}|X^o_{i,j}\right) \right]. \tag{4}$$

The *priors* term includes $p\left(x_0\right)$ and any other available prior information.

In this paper we consider the case of motion (2) and observation (3) models with additive Gaussian noise:

$$x_{i+1} = f\left(x_i, u_i\right) + w_i \quad , \quad w_i \sim N\left(0, \Omega_w\right) \tag{5}$$

$$z_{i,j} = h\left(X^o_{i,j}\right) + v_{i,j} \quad , \quad v_{i,j} \sim N\left(0, \Omega^{ij}_v\right). \tag{6}$$

For notational convenience, we use $\epsilon \sim N(\mu, \Omega)$ to denote a Gaussian random variable ϵ with mean μ and information matrix Ω (inverse of the covariance

matrix). For simplicity we assume the same measurement model $h(\cdot)$ for all observations at time t_i, although the above formulation can be easily generalized.

3.2 Problem Statement

Our goal is to present a general strategy that allows a robot (autonomous vehicle, UAV, manipulator, etc.) planning a suitable control strategy to accomplish a given *task*. Task accomplishment is modeled through an objective function J_k to be optimized; for instance the objective function can penalize distance to a goal position (*path planning*), the uncertainty in the state estimate (*active sensing*), or can model the necessity to visit new areas (*exploration*).

While our formulation is general (only in Section 5 we tailor it to a specific choice of the objective function), we use a motivating example to guide the reader through the derivation. The motivating example is the one of Fig. 2 and includes an *unmanned aerial vehicle* (UAV) flying over an unknown region. The UAV has a downward-looking camera and can observe landmarks in the environment (Fig. 2(a)). The UAV flies in a GPS-denied area and uses the natural landmarks in the environment to estimate its own trajectory and landmarks position. The robot objective is to reach pre-specified goal positions, while preserving an acceptable localization accuracy.
To tackle planning under uncertainty, we adopt a standard *model predictive control* (MPC) strategy in which the robot, at time t_k, has to plan an optimal sequence of controls $u^{\star}_{k:k+L-1} = \{u_k, \ldots, u_{k+L-1}\}$ for L look-ahead steps, so that the objective $J_k(u_{k:k+L-1})$ is minimized over the time horizon (Fig. 4).

In Section 3.2.2 we provide a general formulation of the objective function $J_k(u_{k:k+L-1})$. Before doing that, we need to introduce the concept of *generalized belief,* that encodes the predicted posterior over the robot states and the state of the observed environment.

3.2.1 Generalized Belief Space (GBS)

We define the *generalized belief* at the lth planning step, with $l \in [1, L]$, as

$$gb(X_{k+l}) \doteq p(X_{k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, Z_{k+1:k+l}, u_{k:k+l-1}), \tag{7}$$

where the joint state X_{k+l} includes both the robot states and the observed world states (e.g. landmarks) by time instant $k+l$. In Eq. (7) we separated actions $\mathcal{U}_{k-1}$ and observations $\mathcal{Z}_k$ occurring until the planning time t_k from the actions $u_{k:k+l-1}$ and observations $Z_{k+1:k+l}$ from the first l look-ahead steps (t_{k+1} until t_{k+l}). We note that both $\mathcal{U}_{k-1}$ and $\mathcal{Z}_k$ include the entire history of actions and observations, respectively, until the current time t_k. Additionally, the notation $a_{i:j} \doteq \{a_i, \ldots, a_j\}$ is used. The generalized belief space is nothing more than a standard belief space over a larger state, and we use this terminology to make explicit the fact that our state space also includes the world state.

We represent this belief by a Gaussian

$$gb(X_{k+l}) \sim N\left(X^{\star}_{k+l}, I_{k+l}\right), \tag{8}$$

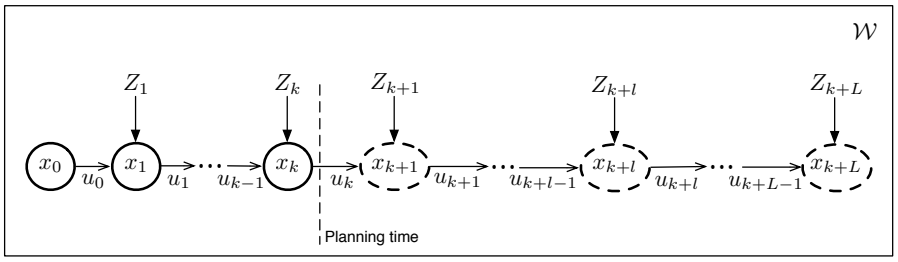


Figure 4: Illustration of the L look-ahead steps planning problem. Past observations $\mathcal{Z}_k = \{Z_0, \ldots, Z_k\}$ and past controls $\mathcal{U}_{k-1} = \{u_0, \cdots, u_{k-1}\}$ are known at the planning time t_k. Future observations $Z_{k+1:k+L}$ are instead unknown and treated as random variables. The objective is to compute a suitable control strategy $u^\star_{k:k+L-1} = \{u_k, \ldots, u_{k+L-1}\}$ for L look-ahead steps. The figure only illustrates the temporal evolution of the system, while each observation may involve generic subset of the states, according to the observation model (3).

where the mean $X^\star_{k+l}$ is set to the maximum a posteriori (MAP) estimate

$$X^\star_{k+l} = \underset{X_{k+l}}{\arg\max}\ gb\left(X_{k+l}\right) = \underset{X_{k+l}}{\arg\min} - \log gb\left(X_{k+l}\right), \tag{9}$$

and the information matrix I_{k+l} is defined accordingly.

We note that while modeling the belief by a single Gaussian is common (see, e.g. (Prentice and Roy, 2009; Platt et al., 2010; Van Den Berg et al., 2012)), it may be not sufficiently descriptive in certain problem domains (e.g. contact, locomotion), in which case mixture of Gaussians is a better representation (see, e.g. (Toussaint, 2009; Erez and Smart, 2010)).

The computation of the parameters (8) that describe the generalized belief is not trivial for two reasons. First, the belief $gb\left(X_{k+l}\right)$ depends on our planning policy, encoded in the future controls $u_{k:k+l-1}$. Second, we do not know a priori which measurements will be acquired at the (future) look-ahead steps. For this reason future observations $Z_{k+1:k+l}$ have to be treated as random variables, as every realization of those variables will lead to a different belief $gb\left(X_{k+l}\right)$. We deal with these aspects in the inner layer of our approach (Section 4.2), while the following section provides a general formulation of the optimal control problem.

3.2.2 Optimal Control Problem

At planning time t_k, the optimal control minimizes an objective $J_k(u_{k:k+L-1})$ for L look-ahead steps. We consider a general objective function

$$J_k\left(u_{k:k+L-1}\right) \doteq \underset{Z_{k+1:k+L}}{\mathbb{E}} \left\{ \sum_{l=0}^{L-1} c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right) + c_L\left(gb\left(X_{k+L}\right)\right) \right\}, \tag{10}$$

that involves L immediate cost functions c_l, one for each look-ahead step. We use the subscript k in J_k to remark that the objective function depends on

the generalized belief $gb\left(X_k\right)$ at planning time t_k. The expectation is taken to account for all the possible observations during the planning lag, since these are not given at planning time and are stochastic in nature (Section 3.2.1).

Each immediate cost function c_l may involve any subset of states $X_{k+l}^c \subseteq X_{k+l}$. The immediate cost c_l can be therefore written as

$$c_l\left(p\left(X_{k+l}^c|\mathcal{Z}_k, \mathcal{U}_{k-1}, Z_{k+1:k+l}, u_{k:k+l-1}\right), u_{k+l}\right). \tag{11}$$

As seen in Eq. (11), the immediate cost function c_l directly involves a distribution over the subset of states X_{k+l}^c. Note that performing inference over the pdf $p\left(X_{k+l}^c|\mathcal{Z}_k, \mathcal{U}_{k-1}, z_{k+1:k+l}, u_{k:k+l-1}\right)$ involves an extended subset of the joint state X_{k+l}: the subset X_{k+l}^c and additional subsets of states X_{k+j}^o that correspond to the future observations $Z_{k+1:k+l}$. For clarity of presentation, however, we proceed in this paper with the pdf over *entire* joint state X_{k+l} that is represented by the belief (7).

The problem addressed in this paper is to find the optimal control policy

$$u_{k:k+L-1}^\star \doteq \left\{u_k^\star, \ldots, u_{k+L-1}^\star\right\} = \underset{u_{k:k+L-1}}{\arg\min}\ J_k\left(u_{k:k+L-1}\right). \tag{12}$$

4 Planning in the GBS

Calculating the optimal control policy (12) involves the optimization of the objective function $J_k\left(u_{k:k+L-1}\right)$. Since the expectation is a linear operator we rewrite the objective function (10) as:

$$J_k\left(u_{k:k+L-1}\right) \doteq \sum_{l=0}^{L-1} \underset{Z_{k+1:k+l}}{\mathbb{E}}\left[c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right)\right] + \underset{Z_{k+1:k+L}}{\mathbb{E}}\left[c_L\left(gb\left(X_{k+L}\right)\right)\right]. \tag{13}$$

According to (13), the objective depends on the (known) generalized belief $gb\left(X_k\right)$ at planning time t_k, on the predicted belief at time $t_{k+1}, \ldots, t_{k+L}$, and on the future controls $u_{k:k+L-1}$. Since in general the immediate costs $c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right)$ are nonlinear functions, it holds

$$\underset{Z_{k+1:k+l}}{\mathbb{E}}\left[c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right)\right] \neq c_l\left(\underset{Z_{k+1:k+l}}{\mathbb{E}}\left[gb\left(X_{k+l}\right)\right], u_{k+l}\right),$$

and we have to preserve the dependence of the generalized belief $gb\left(X_{k+l}\right)$ on the observations $Z_{k+1:k+l}$. Therefore, the belief at the lth look-ahead step depends on $u_{k:k+l-1}$ (which is our optimization variable) and $Z_{k+1:k+l}$.

In order to optimize the objective function (13) we resort to an iterative optimization approach, starting from a known initial guess on the controls. The overall approach can be described as a *dual-layer inference*: the inner layer performs inference to calculate the generalized belief $gb\left(X_{k+l}\right)$ at each of the look-ahead steps, for a given $u_{k:k+L-1}$. The computation of the generalized belief is detailed in Section 4.2 and exploits a computationally efficient technique,

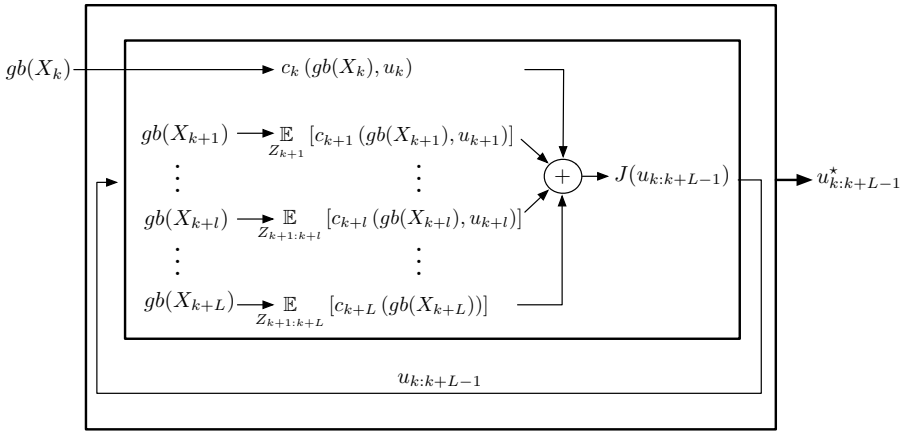


Figure 5: Illustration of the dual-layer inference planning approach. The algorithm takes as an input the GBS at the current time t_k, $gb\left(X_k\right)$, and produces as output a locally-optimal control $u^{\star}_{k:k+L-1}$. The outer layer performs inference over the control $u_{k:k+L-1}$, while the inner layer evaluates the GBS for a given value of $u_{k:k+L-1}$. Note that the GBS at the lth look-ahead step is a function of controls $u_{k:k+l-1}$ as well as of the random observations $Z_{k+1:k+l}$: $gb\left(X_{k+l}\right) = p\left(X_{k+l}|\mathcal{Z}_k, \mathcal{U}_{k-1}, Z_{k+1:k+l}, u_{k:k+l-1}\right)$.

based on *expectation-maximization* and Gauss-Newton method. The outer layer performs inference over the control $u_{k:k+L-1}$, minimizing the objective function (13). The outer layer is discussed in Section 4.1. Since the computation of the optimal control relies on the computation of the generalized belief, we refer to our approach with the term *generalized belief space* (GBS) planning. A schematic representation of the approach is provided in Figure 5.

The presentation of Sections 4.1 and 4.2 is general and does not assume a specific cost function, while in Section 5 we tailor the formulation to a practical choice of the cost function. For this particular choice of the cost we provide algorithmic and implementation details in Section 6.

4.1 Outer Layer: Inference over the Control

Finding a locally-optimal control policy $u^{\star}_{k:k+L-1}$ corresponds to minimizing the general objective function (13):

$$u^{\star}_{k:k+L-1} = \underset{u_{k:k+L-1}}{\arg\min}\, J_k\left(u_{k:k+L-1}\right) \tag{14}$$

The *outer* layer is an iterative optimization over the non-linear objective function $J_k\left(u_{k:k+L-1}\right)$. The optimization starts from a nominal control $u^{(0)}_{k:k+L-1}$, and, at each iteration, computes the delta vector $\Delta u_{k:k+L-1}$ that is used to

update the current values of the controls:

$$u^{(i+1)}_{k:k+L-1} = u^{(i)}_{k:k+L-1} + \Delta u_{k:k+L-1}, \tag{15}$$

where i denotes the iteration number.

According to (13), the objective $J_k\left(u_{k:k+L-1}\right)$ depends on the generalized belief at each look-ahead step, and, in general, calculating $\Delta u_{k:k+L-1}$ involves computing those believes (Section 4.2). The control update (15) is performed in a continuous domain and can be realized using different optimization techniques (e.g., dynamic programming, gradient descent, Gauss-Newton). In this paper we use a simple gradient method to compute $u^{\star}_{k:k+L-1}$.

Gradient Method. At the ith iteration of the gradient method the current control is updated as follows:

$$u^{(i+1)}_{k:k+L-1} = u^{(i)}_{k:k+L-1} - \lambda \ \nabla J_k \tag{16}$$

where ∇J_k is the gradient of the objective function, evaluated at the current guess $u^{(i)}_{k:k+L-1}$, while λ is a suitable *stepsize*. The iterations terminate when a suitable stopping condition is reached. Most common stopping conditions are based on the norm of ∇J_k (the gradient becomes zero at the minimum) or on the relative decrease of the objective at consecutive iterations. Upon convergence, the gradient method provides the optimal control policy $u^{\star}_{k:k+L-1}$ for L look-ahead steps. According to the standard model predictive control framework at each time k the agent solves the optimization problem (14) and only applies the first command $u^{\star}_{k}$, extracted from the computed policy $u^{\star}_{k:k+L-1}$.

4.2 Inner Layer: Inference in GBS

In this section we discuss the computation of the Gaussian approximation (8), whose mean value coincides with the MAP estimate (9). As this inference is performed as part of the higher-level optimization over the control (Section 4.1), the current values for $u_{k:k+l-1}$ are given in the inner inference layer.

There are several complications arising when one tries to solve (9). First, it is unknown ahead of time whether or not a future observation $z_{k+l,j}$ will be obtained: acquiring an observation $z_{k+l,j}$ is a probabilistic event that depends on the true state at time t_{k+l}; for example, if the robot is far away from the jth landmark there will be no measurement $z_{k+l,j}$ in practice. To model this, we introduce a *binary* random variable $\gamma_{i,j}$ for each observation $z_{i,j}$ to represent the event of this measurement being acquired. This is conceptually similar to the Bernoulli random variables used in (Kim and Eustice, 2013) to model the probability of making camera observations based on saliency values.

Second, even if we knew that a given landmark is observed, we cannot know a-priori what the future measurement $z_{i,j}$ will be; hence, we also treat $z_{i,j}$ as

random variable. Therefore, we can define a joint probability density over the random variables in our problem:

$$p\left(X_{k+l}, \Gamma_{k+1:k+l}, Z_{k+1:k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, u_{k:k+l-1}\right), \tag{17}$$

where $\Gamma_i \doteq \{\gamma_{i,j}\}_{j=1}^{n_i}$, with n_i being the number of possible observations at time t_i. In our mobile robotics example, n_i is simply the number of landmarks observed thus far. The probability distribution (17) is defined over the robot and world state (included in the vector X_{k+l}), as well as over the random variables $\Gamma_{k+1:k+l}$, and $Z_{k+1:k+l}$. Using the chain rule we write

$$\begin{aligned} &p\left(X_{k+l}, \Gamma_{k+1:k+l}, Z_{k+1:k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, u_{k:k+l-1}\right) = \\ &p\left(X_{k+l}, \Gamma_{k+1:k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, Z_{k+1:k+l}, u_{k:k+l-1}\right) p\left(Z_{k+1:k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, u_{k:k+l-1}\right). \end{aligned} \tag{18}$$

We now take the assumption that the prior $p\left(Z_{k+1:k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, u_{k:k+l-1}\right)$ is uninformative. This assumption is fairly standard in inference, see, e.g., Eq. (2.55) in (Thrun et al., 2005)), and allows rewriting (18) as:

$$\begin{aligned} p\left(X_{k+l}, \Gamma_{k+1:k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, Z_{k+1:k+l}, u_{k:k+l-1}\right) \propto \\ p\left(X_{k+l}, \Gamma_{k+1:k+l}, Z_{k+1:k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, u_{k:k+l-1}\right). \end{aligned} \tag{19}$$

Since we want to compute the belief $gb\left(X_{k+l}\right)$ in (7), which is defined only over the states, we marginalize the latent variables $\Gamma_{k+1:k+l}$ and get

$$\begin{aligned} gb\left(X_{k+l}\right) = p\left(X_{k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, Z_{k+1:k+l}, u_{k:k+l-1}\right) = \\ \sum_{\Gamma_{k+1:k+l}} p\left(X_{k+l}, \Gamma_{k+1:k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, Z_{k+1:k+l}, u_{k:k+l-1}\right). \end{aligned} \tag{20}$$

Eq. (20) provides an expression for the generalized belief that can be used to compute the MAP estimator in (9). However, two issues are still on the way. First, in order to obtain the distribution $gb\left(X_{k+l}\right)$ one has to marginalize the latent variables $\Gamma_{k+1:k+l}$, according to (20), and this operation is intractable in general. Second, contrarily to standard estimation problems (in which one would resort to numerical optimization techniques to solve (9)) the predicted belief $gb\left(X_{k+l}\right)$ is a function of $Z_{k+1:k+l}$, which are unknown at planning time. Therefore, rather than computing a single vector representing the MAP estimate, here the objective is to compute a vector-valued function of $Z_{k+1:k+l}$, which gives the MAP estimate for each possible value of $Z_{k+1:k+l}$. We solve these two complications exploiting two tools: *expectation-maximization* and a Gauss-Newton approach.

According to the *expectation-maximization* (EM) approach (Minka, 1998), instead of minimizing directly the probability in (9) (which requires to marginalize $\Gamma_{k+1:k+l}$), we minimize the following upper bound of $-\log gb\left(X_{k+l}\right)$:

$$X_{k+l}^{\star} = \arg\min_{X_{k+l}} \mathbb{E}_{\Gamma_{k+1:k+l} | \bar{X}_{k+l}} [-\log p\left(X_{k+l}, \Gamma_{k+1:k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, Z_{k+1:k+l}, u_{k:k+l-1}\right)], \tag{21}$$

where the expectation is computed assuming a given nominal state $\bar{X}_{k+l}$. EM guarantees convergence to a local minimum of the cost in (9), see e.g., (Minka, 1998) and the references therein. We now want to compute explicit expressions for the expectation in Eq. (21). The joint pdf over $X_{k+l}, \Gamma_{k+1:k+l}$ can be written as

$$p\left(X_{k+l},\Gamma_{k+1:k+l}|\mathcal{Z}_k,\mathcal{U}_{k-1},Z_{k+1:k+l},u_{k:k+l-1}\right) \propto \\ p(X_k|\mathcal{Z}_k,\mathcal{U}_{k-1})\prod_{i=1}^{l} p\left(x_{k+i}|x_{k+i-1},u_{k+i-1}\right) p\left(Z_{k+i},\Gamma_{k+i}|X^o_{k+i}\right) \quad (22)$$

and the measurement likelihood term $p\left(Z_{k+i},\Gamma_{k+i}|X^o_{k+i}\right)$ at each look-ahead step i can be further expanded as

$$\prod_{j=1}^{n_i} p\left(z_{k+i,j},\gamma_{k+i,j}|X^o_{k+i,j}\right) = \prod_{j=1}^{n_i} p\left(z_{k+i,j}|X^o_{k+i,j},\gamma_{k+i,j}\right) p\left(\gamma_{k+i,j}|X^o_{k+i,j}\right). \quad (23)$$

Plugging Eqs. (22)-(23) into Eq. (21), recalling the Gaussian motion and observation models (5)-(6), and taking the expectation, results in

$$X^\star_{k+l} = \arg\min_{X_{k+l}} \|X_k - X^\star_k\|^2_{I_k} + \sum_{i=1}^{l} \|x_{k+i} - f\left(x_{k+i-1},u_{k+i-1}\right)\|^2_{\Omega_w} + \\ \sum_{i=1}^{l}\sum_{j=1}^{n_i} p\left(\gamma_{k+i,j}=1|\bar{X}_{k+l}\right) \left\|z_{k+i,j} - h\left(X^o_{k+i,j}\right)\right\|^2_{\Omega^{ij}_v}, \quad (24)$$

where we used the standard notation $\|y-\mu\|^2_\Omega = (y-\mu)^T \Omega (y-\mu)$ for the Mahalanobis norm, with Ω being the information matrix. The first summand in Eq. (24) represents the (known) Gaussian approximation of the generalized belief at the current time: $p(X_k|\mathcal{Z}_k,\mathcal{U}_{k-1}) \sim N\left(X^\star_k, I_k\right)$. The second and third terms describe the influence of future controls and measurements on the belief. In (24), we exploited the fact that the latent variable $\gamma_{k+i,j}$ is binary and, by definition, no observation is taken for $\gamma_{k+i,j} = 0$. The expression $p\left(\gamma_{k+i,j}=1|\bar{X}_{k+l}\right)$ represents the probability of acquiring measurement j at time t_i, assuming that the state is $\bar{X}_{k+l}$: for instance, one can assume that the probability of observing a landmark j decreases with the distance of the robot from the landmark (recall that $\bar{X}_{k+l}$ contains estimates of both robot and landmark positions) and becomes zero outside a given *sensing* radius. Additionally, one can consider only landmark observations within the camera field of view. If we define

$$\bar{\Omega}^{ij}_v = p\left(\gamma_{k+i,j}=1|\bar{X}_{k+l}\right)\Omega^{ij}_v, \quad (25)$$

we can rewrite (24) as:

$$X^{\star}_{k+l} = \underset{X_{k+l}}{\arg\min} \left\| X_k - X^{\star}_k \right\|^2_{I_k} + \sum_{i=1}^{l} \left\| x_{k+i} - f\left(x_{k+i-1}, u_{k+i-1}\right) \right\|^2_{\Omega_w} + \sum_{i=1}^{l} \sum_{j=1}^{n_i} \left\| z_{k+i,j} - h\left(X^o_{k+i,j}\right) \right\|^2_{\bar{\Omega}^{ij}_v}, \quad (26)$$

which suggests the following interpretation: taking the expectation over the binary variables produces a *scaling* effect over the measurement information matrices Ω^{ij}_v; in particular, according to Eq. (25), a low probability $p\left(\gamma_{k+i,j} = 1 | \bar{X}_{k+l}\right)$ is naturally modeled in EM by decreasing the information content of the measurement. In the limit case $p\left(\gamma_{k+i,j} = 1 | \bar{X}_{k+l}\right) = 0$, the matrix $\bar{\Omega}^{ij}_v$ only contains zeros, and the term $\left\| z_{k+i,j} - h\left(X^o_{k+i,j}\right) \right\|^2_{\bar{\Omega}^{ij}_v}$ disappears from the sum, correctly modeling that if $p\left(\gamma_{k+i,j} = 1 | \bar{X}_{k+l}\right) = 0$, then measurement $z_{k+i,j}$ is not actually acquired.

The EM framework allowed to transform the original problem (9) into (26). However, problem (26) is still difficult to solve, since, in general, $f(\cdot)$ and $h(\cdot)$ are nonlinear functions. In estimation, a standard way to solve the minimization problem (26) is the Gauss-Newton method, where a single iteration involves linearizing the above equation about the current estimate $\bar{X}_{l+l}$, calculating the delta vector ΔX_{k+l} and updating the estimate $\bar{X}_{l+l} \leftarrow \bar{X}_{l+l} + \Delta X_{k+l}$. This process should be repeated until convergence. While this is standard practice in information fusion, what makes it challenging in the context of planning is that the observations $Z_{k+1:k+l}$ are unknown and considered as random variables.

In order to perform a Gauss-Newton iteration on (26), we linearize the motion and observation models in Eqs. (5) and (6) about the linearization point $\bar{X}_{k+l}\left(u_{k:k+l-1}\right)$. The linearization point $\bar{X}_{k+l}\left(u_{k:k+l-1}\right)$ is computed as follows. The subset of past states (until time t_k) in $\bar{X}_{k+l}\left(u_{k:k+l-1}\right)$ is set to $X^{\star}_k$, while the future states are predicted via the motion model (6) using the current values of the controls $u_{k:k+l-1}$:

$$\bar{X}_{k+l}\left(u_{k:k+l-1}\right) \equiv \begin{pmatrix} \bar{X}_k \\ \bar{x}_{k+1} \\ \bar{x}_{k+2} \\ \vdots \\ \bar{x}_{k+l} \end{pmatrix} \doteq \begin{pmatrix} X^{\star}_k \\ f\left(x^{\star}_k, u_k\right) \\ f\left(\bar{x}_{k+1}, u_{k+1}\right) \\ \vdots \\ f\left(\bar{x}_{k+l-1}, u_{k+l-1}\right) \end{pmatrix}. \quad (27)$$

One may notice that the linearization point $\bar{X}_{k+l}\left(u_{k:k+l-1}\right)$ is simply the prior at time t_{k+l}, which is the state estimate before including measurements.

Using this linearization point, Eq. (26) turns into:

$$\underset{\Delta X_{k+l}}{\arg\min} \quad \left\| \Delta X_k \right\|^2_{I_k} + \sum_{i=1}^{l} \left\| \Delta x_{k+i} - F_i \Delta x_{k+i-1} - b^f_i \right\|^2_{\Omega_w} + \sum_{i=1}^{l} \sum_{j=1}^{n_i} \left\| H_{i,j} \Delta X^o_{k+i,j} - b^h_{i,j}\left(z_{k+i,j}\right) \right\|^2_{\bar{\Omega}^{ij}_v}, \quad (28)$$

where the Jacobian matrices $F_i \doteq \nabla_x f$ and $H_{i,j} \doteq \nabla_x h$ are evaluated about $\bar{X}_{k+l}\left(u_{k:k+l-1}\right)$. The right hand side vectors b_i^f and $b_{i,j}^h$ are defined as

$$b_i^f \doteq f\left(\bar{x}_{k+i-1}, u_{k+i-1}\right) - \bar{x}_{k+i} \ , \tag{29}$$

$$b_{i,j}^h\left(z_{k+i,j}\right) \doteq z_{k+i,j} - h\left(\bar{X}_{k+i,j}^o\right) \tag{30}$$

Note that $b_{i,j}^h$ is a function of the random variable $z_{k+i,j}$. Also note that under the maximum-likelihood assumption this terms would be nullified: assuming maximum likelihood measurements essentially means assuming zero *innovation*, and $b_{i,j}^h$ is exactly the innovation for measurement $z_{k+i,j}$. We instead keep, for now, the observation $z_{k+i,j}$ as a variable and we will compute the expectation over this random variable only when evaluating the objective function (13). In order to calculate the update vectors ΔX_k and $\Delta x_{k+1}, \ldots, \Delta x_{k+l}$, it is convenient to write Eq. (28) in compact matrix notation:

$$\left\| \mathcal{A}_{k+l}(u_{k:k+l-1}) \ \Delta X_{k+l} \ - \ \breve{b}_{k+l}(u_{k:k+l-1}, Z_{k+1:k+l}) \right\|^2, \tag{31}$$

where we used the relation $\|a\|_\Omega^2 \equiv \left\| \Omega^{\frac{1}{2}} a \right\|^2$, and $\mathcal{A}_{k+l}$ and $\breve{b}_{k+l}$ are of the following form:

$$\mathcal{A}_{k+l} \doteq \begin{bmatrix} \begin{bmatrix} I_k^{1/2} & 0 \end{bmatrix} \\ \mathcal{F}_{k+l} \\ \mathcal{H}_{k+l} \end{bmatrix} \quad , \quad \breve{b}_{k+l} = \begin{pmatrix} 0 \\ \breve{b}_{k+l}^f \\ \breve{b}_{k+l}^h \end{pmatrix}. \tag{32}$$

Here, $\mathcal{F}_{k+l}$ and $\mathcal{H}_{k+l}$ include the Jacobian-related entries $\Omega_w^{\frac{1}{2}} F_i$ and $\left(\bar{\Omega}_v^{i,j}\right)^{\frac{1}{2}} H_{i,j}$ (for all $i \in [1, l]$ and $j \in [1, n_i]$), respectively; in particular:

$$\mathcal{H}_{k+l} = \begin{bmatrix} \ddots & & 0 \\ & \left(\bar{\Omega}_v^{i,j}\right)^{\frac{1}{2}} & \\ 0 & & \ddots \end{bmatrix} H_{k+l} \doteq \breve{\Omega}_v^{\frac{1}{2}} H_{k+l} \tag{33}$$

where H_{k+l} is obtained by stacking measurement Jacobians $H_{i,j}$, with zero blocks for padding. The matrix $\mathcal{H}_{k+l}$ corresponds to the whitened Jacobian of the measurement model (for all measurements). Likewise, the matrix $\mathcal{F}_{k+l}$ is the whitened Jacobian of the motion model for all look-ahead steps; the vectors $\breve{b}_{k+l}^f$ and $\breve{b}_{k+l}^h$ respectively collect the terms $\Omega_w^{\frac{1}{2}} b_i^f$ and $\left(\bar{\Omega}_v^{i,j}\right)^{\frac{1}{2}} b_{i,j}^h(z_{k+i,j})$. The term $\begin{bmatrix} I_k^{\frac{1}{2}} & 0 \end{bmatrix}$ includes a matrix of zeros of appropriate size for padding.

The update vector ΔX_{k+l}, that minimizes (31), is given by

$$\Delta X_{k+l}(u_{k:k+l-1}, Z_{k+1:k+l}) \doteq \left(\mathcal{A}_{k+l}^T \mathcal{A}_{k+l}\right)^{-1} \mathcal{A}_{k+l}^T \breve{b}_{k+l}. \tag{34}$$

Using the vector ΔX_{k+l} in (34) we can update the nominal state $\bar{X}_{k+l}$:

$$\hat{X}_{k+l}\left(u_{k:k+l-1}, Z_{k+1:k+l}\right) = \bar{X}_{k+l} + \Delta X_{k+l} \tag{35}$$

The estimate $\hat{X}_{k+l}\left(u_{k:k+l-1}, Z_{k+1:k+l}\right)$ is the outcome of a single Gauss-Newton iteration on the nonlinear problem (26). We can also compute a local approximation of the information matrix of the estimate as:

$$I_{k+l}\left(u_{k:k+l-1}\right) \doteq \mathcal{A}_{k+l}^{T}\mathcal{A}_{k+l}. \tag{36}$$

We note that in Eq. (34) the (random) measurements $z_{k+l,j}$ only appear in $\breve{b}_{k+l}$. Moreover, since $\breve{b}_{k+l}$ is obtained by stacking vectors b_i^f and $b_{i,j}^h$, and observing that each $b_{i,j}^h$ is a linear function of the corresponding measurement $z_{k+l,j}$, it follows that each entry in $\breve{b}_{k+l}$ is a *linear* function of the measurements $Z_{k+1:k+l}$. This implies that $\Delta X_{k+l}\left(u_{k:k+l-1}, Z_{k+1:k+l}\right)$ and $\hat{X}_{k+l}\left(u_{k:k+l-1}, Z_{k+1:k+l}\right)$ are also linear functions of the measurements. This fact greatly helps when taking the expectation over $Z_{k+1:k+l}$ of the immediate cost function (13). Considering more iterations would better capture the dependence of the estimate on the measurements; however, more iterations would make $\hat{X}_{k+l}\left(u_{k:k+l-1}, Z_{k+1:k+l}\right)$ a nonlinear function of the measurements, making it challenging to devise explicit expressions for (13). We currently assume a single iteration sufficiently captures the effect of the measurements on the generalized belief and we approximate:

$$X_{k+l}^{\star} = \hat{X}_{k+l}(u_{k:k+l-1}, Z_{k+1:k+l}) = \bar{X}_{k+l} + \left(\mathcal{A}_{k+l}^{T}\mathcal{A}_{k+l}\right)^{-1}\mathcal{A}_{k+l}^{T}\breve{b}_{k+l}, \tag{37}$$

Notice that $X_{k+l}^{\star} = X_{k+l}^{\star}(u_{k:k+l-1}, Z_{k+1:k+l})$, i.e., the predicted belief is function of future controls and measurements. Thus, according to the derivation in this section we are now able to compute the predicted belief at the lth look-ahead step, which is parametrized as a Gaussian with mean (37) and information matrix (36).

5 A Specific Family of Objective Functions

The exposition thus far has been given for general immediate cost functions. In this section we tailor our planning approach to the application scenario of Section 3.2 and Fig. 2. In this scenario, the state X_k includes robot trajectory (collection of poses), and 3D positions of landmarks in the environment. At each time step, the UAV has three main objectives: (i) it has to reach a given goal position X^G, (ii) it has to keep its position estimate uncertainty below a given bound β (we will be more formal on this point later), and (iii) it has to minimize control usage. According to these requirements we design the following family of immediate costs:

$$c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right) \doteq tr\left(M_\Sigma I_{k+l}^{-1} M_\Sigma^T\right) + \left\|\zeta\left(u_{k+l}\right)\right\|_{M_u}^2 \tag{38}$$

$$c_L\left(gb\left(X_{k+L}\right)\right) \doteq \left\|E_{k+L}^{G} X_{k+L}^{\star} - X^G\right\|_{M_x}^2 + tr\left(M_\Sigma I_{k+L}^{-1} M_\Sigma^T\right). \tag{39}$$

Here M_Σ, M_u and M_x are weight matrices, and $\zeta(u)$ is some known function that, depending on the application, quantifies the usage of control u. The matrix M_Σ can be also thought as selection matrix: for instance, in our application, we

design the sparsity pattern of M_Σ, such that $M_\Sigma I_{k+l}^{-1} M_\Sigma^T$ returns the marginal covariance of robot position[1] at time t_{k+l} (recall that I_{k+l}^{-1} is the posterior covariance, describing the joint state of robot poses and landmark positions). Therefore, the trace $tr\left(M_\Sigma I_{k+l}^{-1} M_\Sigma^T\right)$ quantifies the amount of uncertainty in robot position at time t_{k+l}. Similarly, E_{k+l}^G is a selection matrix, such that $E_{k+l}^G X_{k+L}^\star$ contains a subset of states for which we want to impose a goal. In our case, this matrix extracts the terminal pose $x_{k+L}^\star$ (at the end of the horizon) from the joint state $X_{k+L}^\star$. Therefore, the term $\left\|E_{k+L}^G X_{k+L}^\star - X^G\right\|_{M_x}^2$ penalizes the distance of the terminal robot position from the goal.

Plugging Eqs. (38) and (39) into Eq. (13), and rearranging the terms, we get

$$J_k\left(u_{k:k+L-1}\right) \doteq \sum_{l=0}^{L-1} \left\|\zeta\left(u_{k+l}\right)\right\|_{M_u}^2 + \sum_{l=0}^{L} tr\left(M_\Sigma I_{k+l}^{-1} M_\Sigma^T\right) + \mathop{\mathbb{E}}_{Z_{k+1:k+L}}\left[\left\|E_{k+L}^G X_{k+L}^\star - X^G\right\|_{M_x}^2\right], \quad (40)$$

where we removed the expectation from the first and the second summand as they do not depend on $Z_{k+1:k+L}$ (future observations), while we maintained it for the last term, as $X_{k+L}^\star$ is a function of $Z_{k+1:k+L}$ according to Section 4.2.

In order to obtain the final expression of the objective function we have to compute the expectation in the last summand in the above equation. This step leads to the following result (see Appendix B for a complete derivation):

$$J_k\left(u_{k:k+L-1}\right) \doteq \overbrace{\sum_{l=0}^{L-1} \left\|\zeta\left(u_{k+l}\right)\right\|_{M_u}^2}^{(a)} + \overbrace{\sum_{l=0}^{L} tr\left(M_\Sigma I_{k+l}^{-1} M_\Sigma^T\right)}^{(b)} + \underbrace{\left[\left\|E_{k+L}^G \bar{X}_{k+L} - X^G\right\|_{M_x}^2 + tr\left(Q_{k+L}\left(H_{k+L} \bar{I}_{k+L}^{-1} H_{k+L}^T + \Omega_v^{-1}\right)\right)\right]}_{(c)}, \quad (41)$$

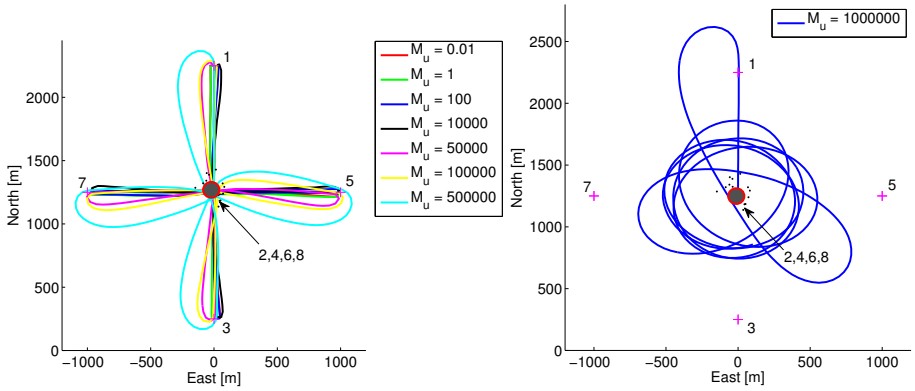
where $\bar{X}_{k+L}$ is the nominal state (27), Ω_v is a block-diagonal matrix containing Ω_v^{ij} on its diagonal, and

$$Q_{k+L} = \left(E_{k+L}^G I_{k+L}^{-1} \mathcal{H}_{k+L}^T \breve{\Omega}_v^{\frac{1}{2}}\right)^T M_x \left(E_{k+L}^G I_{k+L}^{-1} \mathcal{H}_{k+L}^T \breve{\Omega}_v^{\frac{1}{2}}\right). \quad (42)$$

The matrix $\bar{I}_{k+l}$ in Eq. (41) is the information matrix of the nominal state $\bar{X}_{k+l}$, which can be calculated following a similar procedure to calculation of $\bar{I}_{k+l}$ (see Eq. (36)). Specifically, we define the augmented Jacobian matrix $\bar{\mathcal{A}}_{k+l}$ as

$$\bar{\mathcal{A}}_{k+l} \doteq \begin{bmatrix} \begin{bmatrix} I_k^{1/2} & 0 \end{bmatrix} \\ \mathcal{F}_{k+l} \end{bmatrix}, \quad (43)$$

[1]The approach can be easily adapted to minimize map uncertainty. In this case, $M_\Sigma I_{k+l}^{-1} M_\Sigma^T$ would extract landmark marginal covariances from the joint posterior.



(a) (b)

Figure 6: (a) Different M_u values in a basic scenario; (b) Degenerate case for large value of M_u.

which is similar to $\mathcal{A}_{k+l}$ from Eq. (32) but does not include any observations. The a priori information matrix $\bar{I}_{k+l}$ is then given by

$$\bar{I}_{k+l} \doteq \bar{\mathcal{A}}_{k+l}^T \bar{\mathcal{A}}_{k+l}. \tag{44}$$

The matrices $\mathcal{H}_{k+L}$ and $\breve{\Omega}_v$ are defined in Eq. (33). Note that I_{k+l} in Eqs. (41)-(42) represents the updated information matrix, according to Eq. (36), and should not be confused with $\bar{I}_{k+l}$, which is the prior covariance, i.e., before measurement update.

The term (a) in Eq. (41) contains terms penalizing control usage; the term (b) contains terms penalizing uncertainty (captured by the information matrix I_{k+l}). The term (c) was derived from $\underset{Z_{k+1:k+L}}{\mathbb{E}} \left[\left\| E_{k+L}^G X_{k+L}^\star - X^G \right\|_{M_x}^2 \right]$ in Eq. (40), and represents the expected incentive in reaching the goal. We conclude this section by noticing that the term

$$tr\left(Q_{k+L}\left(H_{k+L}\bar{I}_{k+L}^{-1}H_{k+L}^T + \Omega_v^{-1}\right)\right), \tag{45}$$

in Eq. (40), appears because we did not assume maximum likelihood observations. In particular, the term $H_{k+L}\bar{I}_{k+L}^{-1}H_{k+L}^T + \Omega_v^{-1}$ can be easily identified as the innovation covariance; the matrix Q_{k+L} is function of the joint covariance I_{k+L}^{-1}, hence this term rewards uncertainty reduction via loop closures.

6 Implementation Details

6.1 Choice of the Weight Matrices

In this section we discuss how to properly choose the weight matrices M_u, M_Σ, and M_x. Most related work assume these matrices are given, while in practice

their choice can be scenario dependent and can largely influence the control policy.

6.1.1 Choice of M_u

The matrix M_u, appearing in the summand (a) of (41) has a very intuitive function: a larger M_u induces more conservative policies that penalize large controls (or large variations in the controls, depending on the definition of $\zeta(u)$). Consequently, M_u can be tuned to have smoother trajectories or when it is important to keep controls small (e.g., in presence of fuel/power constraints). In our scenario, the control usage $\zeta(u)$ quantifies the change in the yaw angle, which is used by the UAV to control the direction of its motion.

This is demonstrated in Fig. 6a, where we consider a basic scenario in which the robot needs to travel to different goals, each time returning to its starting position (denoted by a filled circle mark). One can observe that higher values of M_u result in smoother trajectories with larger turn radius.

We notice that for large M_u one can incur in degenerate situations (Fig. 6b), in which minimizing control usage interferes with the objective of reaching the goal. After reaching the first goal, the robot should be heading back to its starting position (denoted as the second goal), yet it is stuck and continues performing circle maneuvers, as the attraction to the goal is perfectly balanced by the cost of changing the yaw angle.

In our implementation, we set $M_u = 0.1$: this is a relatively small value that assures that the control penalty does not interfere with the other terms in (41).

In Section 8.1 we further discuss difficulties in tuning weight matrices and we show that, even in a simplified example, using fixed weight matrices may not ensure the desired behavior.

6.1.2 Choice of M_x and M_Σ

The choice of the matrices M_x and M_Σ is less intuitive. A balance between these two matrices is crucial for letting the robot satisfy the concurrent tasks of reaching a goal and minimizing the estimation uncertainty. In this section, we propose a grounded way to select these matrices. In particular, we exploit the fact that, in our application scenario, we are given an upper bound β on the admissible uncertainty, i.e. our planner should be able to impose a soft constraint of the type $tr\left(\bar{M}_\Sigma I_{k+L}^{-1} \bar{M}_\Sigma^T\right) \leq \beta$.

We write M_x and M_Σ as $M_x = \alpha_k \bar{M}_x$ and $M_\Sigma = \sqrt{1-\alpha_k}\bar{M}_\Sigma$, where $\bar{M}_x$ and $\bar{M}_\Sigma$ are selection matrices (only include zero or identity blocks), and $\alpha_k \in [0,1]$ is a scalar weight. $\bar{M}_x$ only "extracts" the final robot position from $\bar{X}_{k+L}$, while $\bar{M}_\Sigma$ selects robot position marginal covariance from I_{k+l}^{-1}. Under these assumptions the objective function becomes:

$$J_k\left(u_{k:k+L-1}\right) = \sum_{l=0}^{L-1} \left\|\zeta\left(u_{k+l}\right)\right\|^2_{M_u} + \alpha_k \sum_{l=0}^{L} tr\left(\bar{M}_\Sigma I^{-1}_{k+l} \bar{M}^T_\Sigma\right) +$$
$$+ \left(1-\alpha_k\right)\left[\left\|E^G_{k+L}\bar{X}_{k+L} - X^G\right\|^2_{\bar{M}_x} + tr\left(Q_{k+L}\left(H_{k+L}\bar{I}^{-1}_{k+L}H^T_{k+L} + \Omega^{-1}_v\right)\right)\right] \tag{46}$$

where α_k controls the balance between the last two terms in the objective (uncertainty reduction versus goal achievement). Here, we want to design a suitable weight α_k, that induces the planner to satisfy the soft bound $tr\left(\bar{M}_\Sigma I^{-1}_{k+L}\bar{M}^T_\Sigma\right) \leq \beta$. The basic intuition is the following: when the uncertainty $tr\left(\bar{M}_\Sigma I^{-1}_{k+L}\bar{M}^T_\Sigma\right)$ is close to the bound β, the robot should prioritize uncertainty reduction, so to avoid the violation of the bound; on the other hand, for small values of $tr\left(\bar{M}_\Sigma I^{-1}_{k+L}\bar{M}^T_\Sigma\right)$, the robot can simply move towards the goal. According to this intuition, we design α_k as

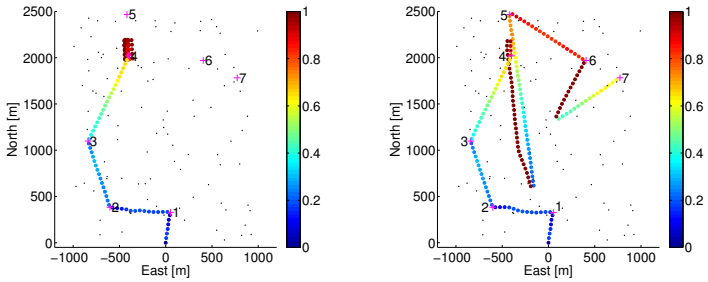
$$\alpha_k = \frac{tr\left(\bar{M}_\Sigma I^{-1}_{k+L}\bar{M}^T_\Sigma\right)}{\beta}, \tag{47}$$

such that for values of $tr\left(\bar{M}_\Sigma I^{-1}_{k+L}\bar{M}^T_\Sigma\right)$ close to the bound β, the ratio α is closer to 1 and the robot will give more importance to the second summand in (46) (i.e., it will prefer minimizing the uncertainty). Conversely, when the uncertainty is far from the upper bound, the term $(1-\alpha_k)$ will be large and the robot will prefer reaching the goal. The a posteriori covariance I^{-1}_{k+L} varies from one control action to another, since different actions may lead the robot to observe different landmarks. This is undesirable, as the weight itself would change depending on the control at which we evaluate the cost. For this reason, in our implementation, we rather use the nominal covariance $\bar{I}^{-1}_{k+L}$ in the computation of α_k. The nominal covariance does not include the effect of loop closures and can be seen as a worst case uncertainty at the end of the horizon. Therefore, in our implementation, we substitute (47) with

$$\alpha_k = \frac{tr\left(\bar{M}_\Sigma \bar{I}^{-1}_{k+L}\bar{M}^T_\Sigma\right)}{\beta}. \tag{48}$$

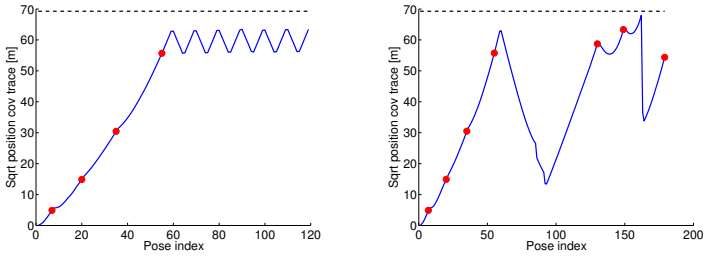
We notice that the quantity $tr\left(\bar{M}_\Sigma \bar{I}^{-1}_{k+L}\bar{M}^T_\Sigma\right)$ can eventually become larger than β, as we are not imposing a hard uncertainty constraints, and for this reason it is convenient to rewrite α_k as

$$\alpha_k = \min\left(\frac{tr\left(\bar{M}_\Sigma \bar{I}^{-1}_{k+L}\bar{M}^T_\Sigma\right)}{\beta}, 1\right). \tag{49}$$



(a) α-colored Trajectory

(b) α-colored Trajectory



(c) Square root of covariance trace

(d) Square root of covariance trace

Figure 7: Balancing uncertainty reduction and exploration: a naïve approach can easily result in oscillator behavior and repeated loop-closures (a) and (c); The proposed techniques avoids this by requiring significant uncertainty reduction before proceeding to goal (b) and (d). (a),(c): robot trajectory; we color each robot position depending on the current value of $\alpha_k \in [0, 1]$. (b),(d): square root of the trace of robot position covariance.

Uncertainty budget. While Eq. (49) provides a mean to prioritize uncertainty reduction, additional considerations are needed to tackle the transition from uncertainty reduction to exploration. In other words, how to determine uncertainty has been sufficiency reduced and it is time to proceed to the goal?

To clarify this issue, we use the example in Fig. 7. The UAV starts at the origin and visits goals 1 to 4. Right after reaching goal 4, the planner realizes that the uncertainty is too high ($\alpha_k \approx 1$) and guides the robots to re-observe landmarks located nearby goal 4. However, since the uncertainty over these landmarks is not significantly smaller than robot's uncertainty, the latter gets only slightly reduced (see trace of covariance in Fig. 7c). As a result, when the robot heads back to the goal, its uncertainty quickly reaches β again and the process repeats itself. Therefore, the robot will never attain the goal, and will be stuck in the oscillatory behavior of Figs. 7a. Consistently with the trajectory, the trace of the covariance (Fig. 7c) keeps oscillating between uncertainty increase (i.e., the robot tries to reach the goal), and uncertainty reduction (i.e., the robot goes back to revisit landmarks).

One can interpret this problem in terms of "uncertainty budget": at each

instant of time the robot has an uncertainty budget, that is the increase of uncertainty the robot can afford before the uncertainty upper bound is reached. Since each motion implies an increase in the uncertainty (in absence of loop closure), the uncertainty budget defines a number of steps the robot can take without closing the loop and still satisfying the bound. Revisiting landmarks can increase this budget, as it reduces robot uncertainty. In particular, visiting landmarks with low uncertainty can significantly increase the uncertainty budget of the robot, as it re-establishes a very small uncertainty on robot position. Conversely, less-certain landmarks reduce only slightly robot uncertainty.

Using the concept of uncertainty budget we can interpret Fig. 7 as follows: the loop closure action does not provide a sufficient increase in the uncertainty budget, and the latter remains insufficient to reach the goal.

In this work we applied a simple heuristic to mitigate this problem. After reaching the value $\alpha_k = 1$, we require a considerable reduction of the uncertainty (i.e., to accumulate a large uncertainty budget), before resetting α_{k+1} to a value smaller than 1. This way, the robot keeps pursuing loop closures until its uncertainty is small. Specifically, we set α_{k+1} to a value smaller than 1 only if $\alpha_{k+1} < \alpha_{\text{LB}}$, where α_{LB} defines a desired lower bound for the uncertainty (i.e., a minimum uncertainty budget). In our implementation we set $\alpha_{\text{LB}} = 0.6$. We found this heuristic allows to cope with most scenarios. For example, the result of applying this heuristic in the previous scenario is shown in Figs. 7b and 7d. After reaching $\alpha_k = 1$, the planner continues to guide the robot to perform loop closure until observing landmarks that are known with good precision. Only then α_k is updated to a value smaller than 1 and the robot proceeds to goal 5.

We will confirm the effectiveness of this heuristic in Section 7. However, it is still possible to devise degenerate cases in which, even this strategy would fail to attain the goal. We further discuss this problem in Section 8.2, where we show that the impossibility to reach the goal within a given uncertainty budget is an intrinsic limitation of the problem, and it is not due to the used technique.

6.2 Algorithmic Implementation

We summarize our GBS planning approach, for the specific application scenario of Section 5, in Algorithms 1-4.

Algorithm 1 discusses inference over the control. This is the outer layer which includes two main blocks. The first is the computation of the weight matrices (lines 14-21), according to Section 6.1.2. Given the weight matrices M_u, M_x and M_Σ, the objective function (41) is uniquely defined, and we can apply the gradient method, which constitutes the second block of Algorithm 1 (lines 22-34). The gradient method (Section 4.1) requires to compute the gradient and evaluate the objective function, and, upon convergence, returns the optimal control $u^\star_{k:k+L-1}$. The computation of the gradient of the objective function (41) is detailed in Algorithm 2. Deriving an analytic expression of the gradient is hard in general, due to the nonlinearity induced by the motion model and the matrix inversion. Therefore, we resort to numerical derivatives, which require evaluating the objective function for small perturbations of the control.

Inputs:
- $X_k^\star, I_k$: parametrization of belief $gb(X_k)$ at current time t_k
- $u_{k:k+L-1}^{(0)}$: nominal control
- β: uncertainty upper bound
- α_{LB}: lower bound for α
- α_{k-1}: scalar weight from previous time step
- M_u, $\bar{M}_x$, $\bar{M}_\Sigma$: weights/selection matrices
- λ: stepsize for the gradient method

Outputs:
- $u_{k:k+L-1}^\star$: optimal control over the horizon lag

Initialize: $i = 0$, $J_k^{(0)} = 0$ `/* Reset objective and iterations */`

set weight matrices:

`/* Calculate` α_k `from a priori covariance` $\bar{I}_{k+L}$ `*/`

$\bar{I}_{k+L} =$ `innerLayer`$(X_k^\star, u_{k:k+L-1}^{(0)})$ (Alg. 4)

$$\alpha_k = \min\left(\frac{tr\left(\bar{M}_\Sigma \bar{I}_{k+L}^{-1} \bar{M}_\Sigma^T\right)}{\beta}, 1\right)$$

if $\alpha_{k-1} == 1$ **and** $\alpha_k > \alpha_{LB}$ **then**
- $\alpha_k = 1$

end

`/* Set weight matrices */`

$$M_x = (1 - \alpha_k)\bar{M}_x, \quad M_\Sigma = \sqrt{\alpha_k}\bar{M}_\Sigma$$

end

while *true* `/* Gradient method */`

do

update $u_{k:k+L-1}^{(i)}$**:**

$\nabla J_k =$ `computeGradient`$(X_k^\star, I_k, u_{k:k+L-1}^{(i)}, M_u, M_x, M_\Sigma)$ (Alg. 2)

$u_{k:k+L-1}^{(i+1)} = u_{k:k+L-1}^{(i)} - \lambda \nabla J_k$

end

Check convergence:

$J_k^{(i+1)} =$ `evaluateObjective`$(X_k^\star, I_k, u_{k:k+L-1}^{(i+1)}, M_u, M_x, M_\Sigma)$ (Alg. 3)

if $\|\nabla J_k\| < \epsilon$ **or** $|(J_k^{(i+1)} - J_k^{(i)})/J_k^{(i+1)}| < \epsilon$ **or** $i \geq i_{max}$ **then**

return $u_{k:k+L-1}^\star = u_{k:k+L-1}^{(i+1)}$

end

end

`/* Move to next iteration */`

$i = i + 1$

end

Algorithm 1: `outerLayer` performs inference over the controls.

Inputs:
  $X_k^\star, I_k$: parametrization of belief $gb(X_k)$ at current time t_k
  $u_{k:k+L-1}$: control actions from time t_k to t_{k+L-1}
  M_u, M_x, M_Σ: weight matrices
Outputs:
  ∇J_k: gradient of the objective function

`/* Evaluate objective function using current controls */`
J_k=`evaluateObjective`$(X_k^\star, I_k, u_{k:k+L-1}, M_u, M_x, M_\Sigma)$ (Alg. 3)

`/* Calculate numerical gradient by */`
`/* perturbing control at each look-ahead step */`
for $l = 0 : L-1$ **do**
  $\delta u = zeros(L, 1)$
  $\delta u(l) = \epsilon$ `/* add small perturbation on the` l`th control */`
  J_k^p=`evaluateObjective`$(X_k^\star, I_k, u_{k:k+L-1} + \delta u, M_u, M_x, M_\Sigma)$ (Alg. 3)
  $\nabla J_k(l) = (J_k^p - J_k)/\epsilon$
end
return ∇J_k

Algorithm 2: `computeGradient` computes the gradient of the objective function J_k.

Inputs:
  $X_k^\star, I_k$: parametrization of belief $gb(X_k)$ at current time t_k
  $u_{k:k+L-1}$: control actions from time t_k to t_{k+L-1}
  M_u, M_x, M_Σ: weight matrices
Outputs:
  J_k: value of the objective function for the control $u_{k:k+L-1}$
`/* Calculate` $gb(X_{k+l})$ `for each look-ahead step */`
for $l = 1 : L$ **do**
  $[\bar{X}_{k+l}, \bar{I}_{k+l}, I_{k+l}]$ = `innerLayer`$(X_k^\star, I_k, u_{k:k+l-1})$ (Alg. 4)
end
Evaluate J_k according to Eq. (41) using $\bar{X}_{k+l}, \bar{I}_{k+l}, I_{k+l}$, $l \in [0, L]$
return J_k

Algorithm 3: `evaluateObjective` computes objective function for a given control.

Inputs:
  $X_k^\star, I_k$: parametrization of $gb(X_k)$ at time t_k
  $u_{k:k+l-1}$: control actions from time t_k to t_{k+l-1}
Outputs:
  $\bar{X}_{k+l}$: nominal estimate at time t_{k+l}
  $\bar{I}_{k+l}$: nominal information matrix at time t_{k+l}
  I_{k+l}: a-posteriori information matrix at time t_{k+l}
Calculate nominal estimate $\bar{X}_{k+l}$ using Eqs. (27)
Calculate nominal information matrix $\bar{I}_{k+l}$ using (44)
Calculate augmented $\mathcal{A}_{k+l}$ and $\breve{b}_{k+l}$ using Eq. (32)
Calculate a-posteriori information matrix I_{k+l} using Eq. (36).
return $\bar{X}_{k+l}$, $\bar{I}_{k+l}$, I_{k+l}

Algorithm 4: Inner Layer: Inference in GBS for lth look-ahead steps.

The computation of the gradient and the outer layer require the evaluation of the objective function for a given control $u_{k:k+L-1}$. This evaluation is described in Algorithm 3. Here, we propagate the generalized belief for L look-ahead steps and use the predicted belief to feed the cost in Eq. (41).

The above algorithms rely on the capability to predict the (known) generalized belief from the current time for L look-ahead steps. The description of how to predict the current belief to the lth look-ahead step is reported in Section 4.2 and summarized in Algorithm 4. We notice that, in our implementation, the inner layer returns both the a-priori and the a-posteriori information matrix, $\bar{I}_{k+L}$ and I_{k+L} , respectively, since they both appear in the objective function.

6.3 Computational Considerations

We let n represent the dimensionality of the state vector X_k at the current time t_k. Assuming X_k consists of k robot poses and m landmarks: $n = 6k + 3m$. In the experiments reported in Section 7, n is in the order of thousands as the considered scenarios include many poses and landmarks. We keep past poses as part of the state vector since it entails sparsity, making the optimization problem underlying MAP estimation (Section 4.2) more efficient (Strasdat et al., 2010).

The outer layer of our approach performs gradient descent optimization, using finite differences to calculate gradient of the objective function (Algorithm 2). Typically, the optimization converges in less than 100 iterations, and requires much less iterations when nominal control is already close to the (locally) optimal value. Each gradient iteration involves the inner layer that performs inference over the generalized belief, at each of the L look ahead steps, given appropriate controls (Algorithm 4). As opposed to (Van Den Berg et al., 2012), we operate in an information form which admits efficient calculations, in particular for the a posteriori information matrix $\mathcal{I}_{k+l}$ (according to Eq. (36)), where sparsity is fully exploited. The most expensive operation is calculating the covariance over variables of interest, as determined by the selection matrix M_Σ, which in the considered scenario comprise only the robot current pose.

However, computational complexity of this operation can be greatly reduced by enforcing appropriate variable ordering, eliminating the current robot pose last (Kaess et al., 2011; Kaess and Dellaert, 2009). In summary, if the dimension of the control u_k is d_u, each gradient computation entails Ld_u calls to the inner layer, as we have to compute finite differences for each control and for each look-ahead step.

Relation with sampling-based approaches. Recent work, such as POMCP by Silver and Veness (Silver and Veness, 2010) and the recent approach by Bai et al. (Bai et al., 2013), showed impressive results on very large POMDP using sampling. In the rest of this section we briefly discuss the use of sampling-based approaches in our application.

Both sampling-based approaches and our approach aim at avoiding the curse of dimensionality (Silver and Veness, 2010). In our formulation, we avoid the curse in inference, by taking a Gaussian belief assumption (Eq. (8)); moreover, we avoid it in planning, by using a local search (gradient method) within a model predictive control framework. POMCP and the approach by Bai et al. use sampling to avoid the curse of dimensionality. However, in the considered problem, the application of sampling-based techniques has two main drawbacks. First, the use of a sampling-based representation would be problematic for inference: we operate over a large (n), and the number of samples needed to have satisfactory results would be prohibitively large. The fact that our objective function includes the posterior covariance further stresses the need of many samples that are required to have a statistically meaningful covariance estimate. For this reason in SLAM, optimization-based techniques appear as the dominant paradigm and are preferred over sampling-based approaches (e.g., Rao-Blackwellized Particle filters). Second, sampling-based planning approaches model possible *histories* as sets of discrete controls and observations (Silver and Veness, 2010): this would not be a natural model for our problem, in which the robot takes (real-valued) distance and bearing measurements from a potentially large number of landmarks, and controls live in a continuous domain. Finally, one of the key advantages of sampling-based approaches (e.g. (Silver and Veness, 2010)) is that simulating possible histories is inexpensive, and one can simulate hundreds of thousands of histories per second. However, in our problem a single simulation entails solving a SLAM problem with hundreds of poses and landmarks, and this is *not* inexpensive. On the other hand, in our approach, we only need $d_u L$ simulations (from the inner layer) at each gradient iteration. Therefore, since we usually have less than 100 gradient iterations and since the dimension of the control d_u is small in the considered application, our approach requires a reasonable number of simulations. The computation can be further improved by resorting to automatic differentiation which can be used to calculate exact gradients without numerical differentiation (Sommer et al., 2013; Patil et al., 2014).

7 Experiments

In this section we present an extensive analysis of the proposed technique and benchmark it against related work.

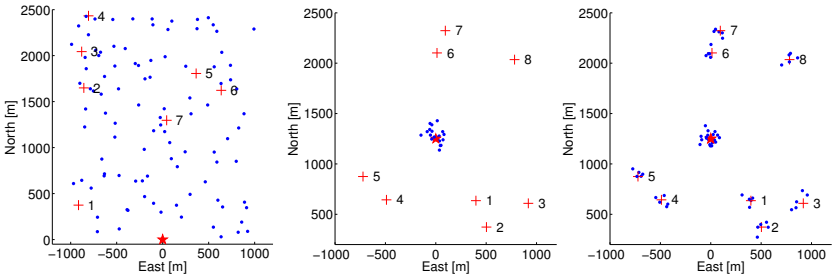
Experimental setup The simulation scenario reflects the motivating example presented at the beginning of this paper. We consider an unmanned aerial vehicle (UAV) that flies at a given height over an initially unknown region (Fig. 2). The UAV has a downward-looking camera and can observe natural landmarks in the environment. The UAV flies in a GPS-denied area and uses these landmarks to estimate its own position. The objective is to reach prespecified goal positions, while preserving an acceptable localization accuracy, quantified by the trace of the covariance of its position estimate.

For our experiments, we assume that the UAV flies at a constant height of 500m above the ground and travels at a constant speed of 50m/s. For simplicity, we assume the robot can only control its yaw angle: The control effort $\zeta(u)$ in Eq. (41) is therefore defined as the change in the yaw angle. In our simulations, we generate landmarks in the environment; these are 3D points, whose positions are randomly drawn (e.g., uniformly on the ground level). Using the onboard camera, the UAV can detect and measure bearing to these landmarks. The image observations are calculated by projecting the ground truth values of 3D points onto the camera plane and corrupting the result with a measurement noise drawn from a zero-mean Gaussian distribution with standard deviation (std) of $\sigma_{\text{pixel}} = 1$ pixel. We assume the camera calibration matrix is known. Additionally, we assume the distance to the observed features can be measured. This information can be obtained either from a range sensor or a stereo camera, or alternatively, it can be calculated assuming the flight height is known. In our simulation the range measurements are calculated from the ground truth distance of the robot to each of the landmarks, corrupted with zero-mean Gaussian noise with std of $\sigma_{\text{range}} = 1$ m.

When deploying the system, a human operator decides a fixed sequence of goal positions the UAV has to reach. As a second requirement, the human operator fixes an upper bound β on the position estimation error, the UAV should guarantee. In our formulation the bound is a soft constraint, as the objective function (41) penalizes violation of such bound, but it does not enforce it; we discuss the use of hard constraints in Section 8.3.

The UAV autonomously plans its motion strategy using our planning approach. Unless otherwise mentioned, the number of look-ahead steps is set to $L = 5$. Additionally, the weight M_u, in the objective function (41) is chosen as $M_u = 0.1$; the matrices M_x and M_Σ are computed as described in Section 6.1.2. The upper uncertainty bound β (see Section 6.1) was set to 69^2 meters squared, which corresponds to a bound of 40 meters in each axis.

Test scenarios We evaluate our approach in 3 scenarios, each with a different landmark distribution (Fig. 8): Uniform, Oasis, Clusters. Landmarks are



(a) (b) (c)

Figure 8: Considered scenarios: (a) UNIFORM. (b) OASIS. (c) CLUSTERS. Robot initial position is denoted by red filled star.

uniformly scattered in the UNIFORM scenario (Fig. 8a), centered around the starting robot location in the OASIS scenario (Fig. 8b), and clustered around each goal position and around the starting robot location in the CLUSTERS scenario (Fig. 8c).

Evaluation metrics We consider the following metrics to analyze performance of our approach and compare it against other techniques.

Covariance trace. We use the trace of the covariance of the current position estimate as uncertainty metric. The position covariance is a 3 by 3 positive definite matrix, having on the diagonal the variances along the Cartesian directions. The upper bound of the trace has a very intuitive meaning: it is the maximum uncertainty we tolerate as sum of the variances along the three axis.

Estimation errors and miss distances. The objective of bounding admissible covariance is to keep estimation errors small. We compare different approaches in terms of estimation errors, which, at each time step, are computed as the distance between the estimated 3D position and the ground truth position of the UAV. In the considered application scenario, a particularly important error is the one that is committed when reaching the goals. For instance, if the robot has to acquire pictures of a target at a given goal position, a large error is undesirable, as it may limit the usability of the collected information. For this reason, we also compare the approaches in terms of *miss distance*, which is the estimation error when a goal is attained.

Trajectory quality and length. Since the UAV moves at constant speed, trajectory length measures the time required by the approach to cover the scenario and reach the assigned goals. We use it as a quantitative measure of how efficient are the paths produced by each technique. A bad planning approach produces non-smooth trajectories with many wasteful segments in which robot motion is not rewarded by uncertainty reduction and goal accomplishment. A good planning approach is efficient in reaching goals in short time, while preserving acceptable uncertainty. We complement this quantitative metric with

qualitative observations of the smoothness of planned trajectories.

Control usage. The control usage is defined as the change in yaw angle at consecutive time steps. This is essentially the term (a) in Eq. (41). A good planning algorithm produces smooth trajectories, in which small yaw changes are preferred over sudden or unreasonable direction changes.

Planning time. We report the time it takes for the different approaches to compute a plan. All approaches are implemented in Matlab, and rely on the GTSAM optimization library[2] as inference engine. The time results are mainly shown to discuss factors influencing computational effort, while the usual rule of thumb is that the timing decreases by at least a factor of 10 when porting the code into a compiled language, e.g., C++.

7.1 Performance Analysis for Planning in GBS

For sake of clarity, in Section 7.1.1, we start with the presentation of a "typical" scenario, which summarizes the most common behaviors we observed in simulations. Then, in Section 7.1.2, we present a statistical study, which demonstrates the robustness or our approach against different noise realizations. In Section 7.1.3, we investigate the use of different planning horizons. Finally, in Section 7.1.4, we demonstrate our method performs well in a large variety of scenarios.

7.1.1 Typical Scenario

The results of this section are obtained in a UNIFORM scenario, with randomly distributed goal locations, as shown in Fig. 9. The trajectory produced by our planning approach is given in Fig. 9a (and the zoomed view of Fig. 9b): the *actual* trajectory of the UAV is shown in blue. The *estimated* trajectory, used by the planner, is shown in red, together with the estimated covariance ellipses. For simplicity we do not show the z component of the covariance, however, it is fully considered in planning. The figure also indicates the estimated landmark coordinates (green '+' marks), and the corresponding covariance ellipses (also in green). Ground truth positions for the landmarks are denoted as blue points.

The robot starts operating at the origin. Initially, it does not know anything about the environment. While traveling towards goal 1, the UAV starts observing landmarks, and builds its world model, as a collection of landmark position estimates. Similarly, more and more landmarks are observed on the way to goals 2, 3, and 4. Observation of new (unknown) landmarks does not improve the estimate of robot position. For this reason, until reaching goal 4, the trace of the position covariance (Fig. 9d) keeps increasing. After reaching goal 4, the covariance trace is very close to the specified upper bound $\beta = 69^2$ m^2, and the planner guides the robot to visit previously-observed landmarks to reduce uncertainty via loop-closure. When heading back to observed landmarks, the covariance trace (Fig. 9d) decreases, and it is eventually re-established to a small value. After the uncertainty reduction, the UAV can head back to the next goal (goal 5), and completes the assigned tasks.

[2] https://collab.cc.gatech.edu/borg/gtsam

As discussed in Section 6.1.2, the parameter α_k plays a key role in balancing uncertainty reduction and distance to goal. The value of this parameter for each time step is shown in Fig. 9c. It is easy to see the correspondence between this figure and Fig. 9a, since larger covariances induce α_k values closer to 1. Moreover, when α_k reaches 1, a sufficient reduction of the uncertainty is required before re-establishing a value smaller than 1 (see Section 6.1.2).

Observe that α_k may reach 1 even though the actual covariance does not cross the bound (Fig. 9d). This is the case since α_k is set according to the a priori uncertainty at the last look-ahead step, see Eq. (49). This allows to predict when the uncertainty bound is about to be reached and to take action, i.e. move towards loop closure, to remedy this situation.

Finally, in Figs. 9e-9f we show, respectively, control effort and cumulative control effort. The former represents the change in heading angle at each time step; the latter is calculated as the sum of absolute values of these changes. Each peak in Fig. 9e corresponds to a sharp turn in the trajectory of Fig. 9a. In the sequel, we will be using cumulative control as one of the performance metrics to compare our approach to other related approaches.

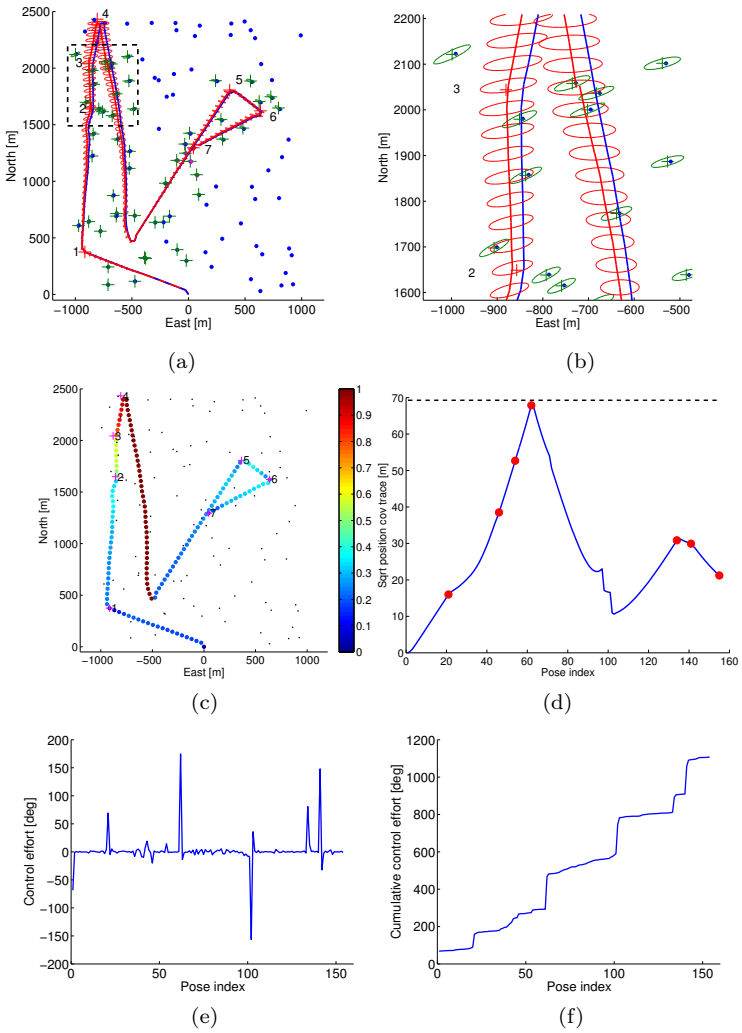
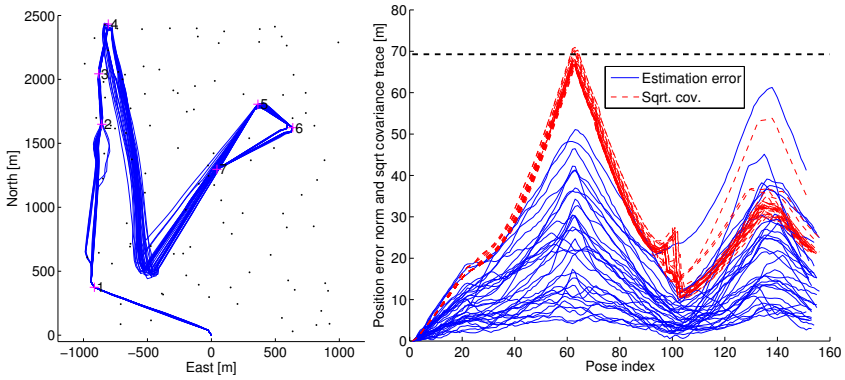


Figure 9: UNIFORM scenario: (a), (b): Ground truth (blue line) and estimated trajectory (red line), with covariance ellipses (red). Ground truth (blue dots) and estimated landmark positions (green+) with confidence covariances (green). (c) Estimated trajectory: the color scale is used to report, for each position, the corresponding value of α_k. (d) Square root of the trace of the position covariance; desired upper bound β is shown as a dashed black line, while the red dots denote the instant of time in which the UAV accomplished a goal. (e) Control effort. (f) Cumulative control effort.



(a) (b)

Figure 10: Statistical study (30 runs) of planning in the GBS approach in UNIFORM scenario. (a) Trajectories; (b) Position error norm and square root of the covariance trace.

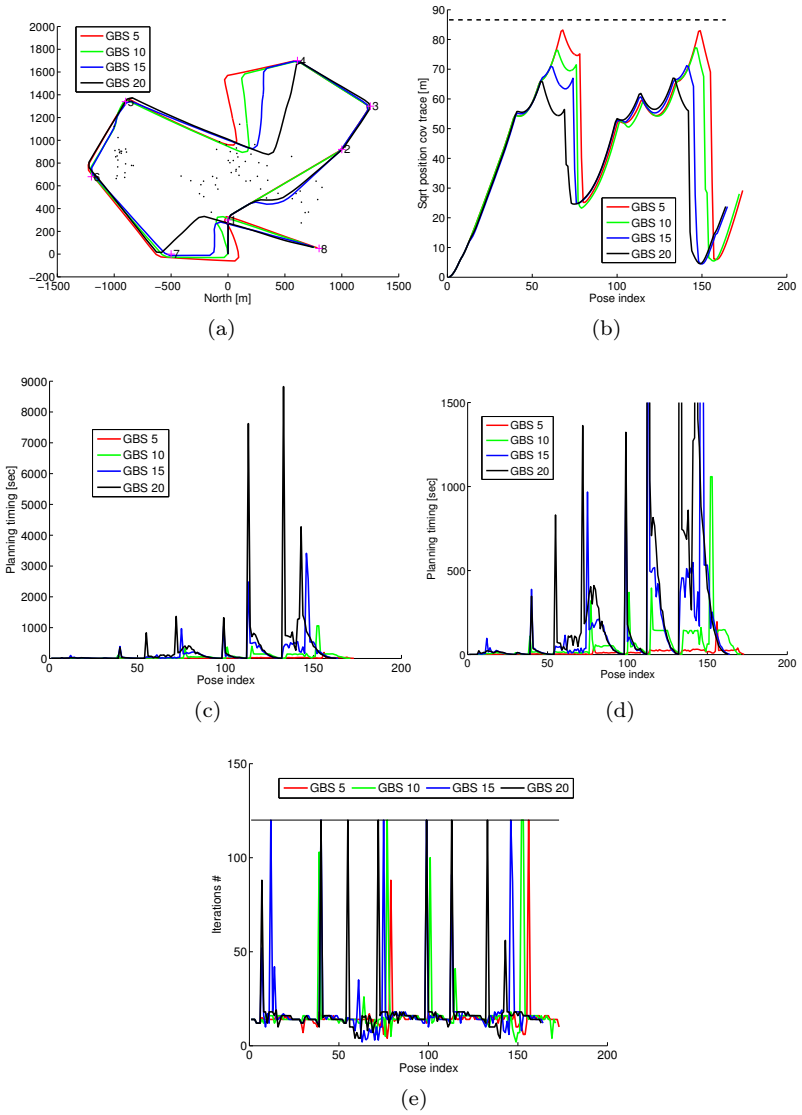
7.1.2 Statistical Study

We investigate the robustness of our approach to different noise realizations in a statistical study for the UNIFORM scenario. The same landmark and goal configuration as in Section 7.1.1 is used, while process and measurement noise were randomly drawn from appropriate distributions for each run. The results of 30 such runs are given in Fig. 10: Fig. 10a shows the trajectories, while Fig. 10b shows the norm of position estimation errors and the square root of the covariance trace. One can observe all trajectories are reasonable and smooth with a similar loop closure event after reaching the forth goal. Estimation errors are always below the covariance, indicating the inference engine is consistent, and the final drop in uncertainty covariance due to loop closure can be clearly seen around pose index 100.

7.1.3 Different Planning Horizon Lags

In this section we study the effect of using different planning horizons L. Figs. 11a and 11b show results for $L = 5, 10, 15$ and 20 look-ahead steps. The scenario comprises 8 randomly generated goals, and landmarks scattered in 4 clusters. Soon after reaching goal 4, the planner guides the robot towards loop closure. Longer horizon lags lead the robot to go earlier towards a loop closure. This is due to the fact that α_k depends on the a-priori uncertainty at the end of the horizon, and this may be pessimistic when the horizon is long. Therefore, the planning approach can be made more conservative by increasing the horizon[3].

[3]One could use a longer horizon just for predicting how soon the uncertainty will cross the bound β and a shorter horizon for actually calculating the control policy. We leave the investigation of this aspect to future work.



(a) (b) (c) (d) (e)

Figure 11: Different horizon lags. (a) Trajectories, (b) Square root of the trace of position covariance, (c) Computational time, (d) Zoom-in on computational time (e) Number of iterations of the gradient method.

Increasing the horizon lag impacts the computational time, as the inner layer has to predict the generalized belief for a larger number of look-ahead steps. Fig. 11c shows that larger horizons entail large (and sometimes impractical) computational cost. For $L = 5$ the computational time is acceptable, as we will remark in the comparison with related work. The spikes observed in

Fig. 11c are related to the number of iterations in the gradient method (outer layer of our approach). Clearly, a larger number of iterations implies a larger computational effort. The number of iterations is reported in Fig. 11e, together with the maximum number of iterations, used as stopping condition in the gradient method (i_{max} in line 29 of Algorithm 1). The planner requires a large number of iterations when it has to apply a large change in the steering angle (this is usually the case after reaching a goal, or after a loop closure). This is due to the fact that we initialize the gradient method with the current control (i.e., the nominal control is the current steering angle); therefore, it takes more iterations to converge to an optimal control that is far from the initial guess.

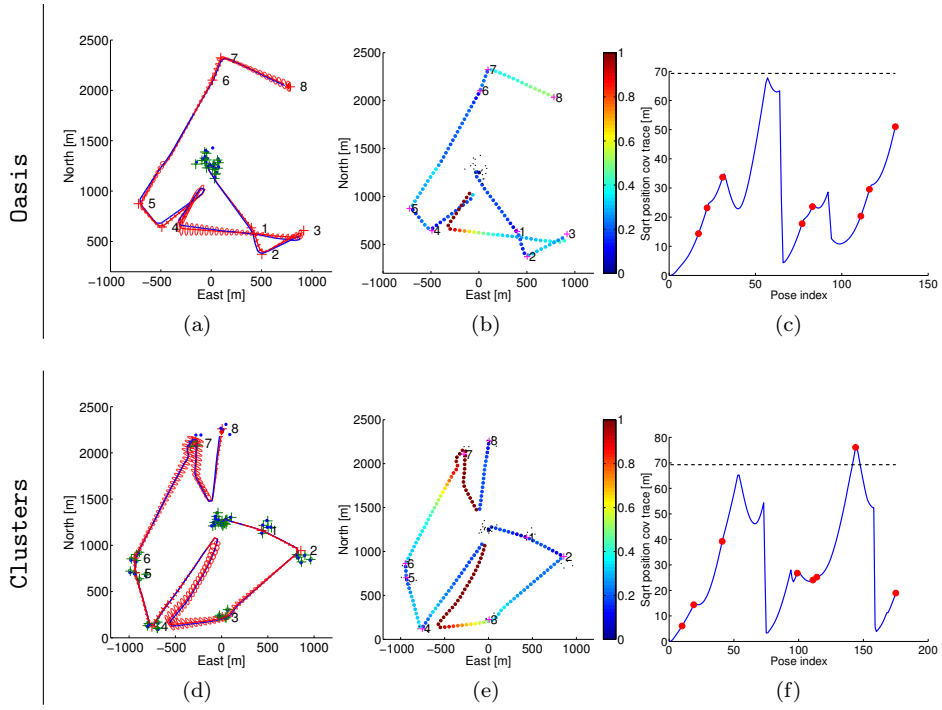


Figure 12: Typical runs for Oasis (first row) and Clusters (second row) scenarios. (a),(d): Trajectories with covariances ellipses; (b),(e): Trajectories color-coded as a function of α_k; (c),(f): Square root of position covariance trace.

7.1.4 Additional Scenarios

In this section we demonstrate the effectiveness of our approach in a large variety of scenarios by considering different random goal configurations in all the three test scenarios: Uniform, Oasis and Clusters.

We already discussed a typical result for the Uniform scenario in Section 7.1.1. The typical outcomes in the Oasis and Clusters scenarios are shown in Fig. 12 using a similar format as in Section 7.1.1: actual and estimated

trajectory with covariances and landmark estimates, color-coded trajectories according to the parameter α_k, and the square root of the position covariance trace. Our approach efficiently attains the goals and preserves an acceptable uncertainty (which is mostly below the bound β), by guiding the robot towards informative loop closures. Specifically, for the CLUSTERS scenario, our approach correctly guides the robot towards the most informative cluster (located in the robot start position), which results in a very large decrease in the uncertainty.

In order to show that our approach exhibits good performance in a variety of goal configurations, we consider five randomly drawn goal positions in each of the three scenarios. The results are given in Figs. 13. Planning in the GBS produces natural trajectories with acceptable uncertainties in all cases.

7.2 Comparison with Related Work

We compare four different techniques.

GBS planning. This is the planning approach proposed in this paper. We set $L = 5$ as time horizon. The weight matrices are computed as in Section 6.1.2 and $M_u = 0.1$.

GBS-ML planning. This is a variation of the GBS approach in which we neglect the last term in Eq. (41). The objective function hence becomes:

$$J_k\left(u_{k:k+L-1}\right) = \sum_{l=0}^{L-1} \left\|\zeta\left(u_{k+l}\right)\right\|^2_{M_u} + \sum_{l=0}^{L} tr\left(M_\Sigma I^{-1}_{k+l} M^T_\Sigma\right) + \left\|E^G_{k+L}\bar{X}_{k+L} - X^G\right\|^2_{M_x}. \tag{50}$$

Neglecting this term essentially means assuming maximum likelihood observations, as the term is the outcome of the expectation over the measurements. We consider this technique to understand the impact of the maximum likelihood assumption on planning performance. The horizon length and the weight matrices are set as in the GBS approach.

Continuous planning with no uncertainty (CNU). This technique neglects the uncertainty in the belief and only attempts to minimize the distance to the goal and the control effort. The objective function (41) therefore becomes:

$$J_k\left(u_{k:k+L-1}\right) = \sum_{l=0}^{L-1} \left\|\zeta\left(u_{k+l}\right)\right\|^2_{M_u} + \left\|E^G_{k+L}\bar{X}_{k+L} - X^G\right\|^2, \tag{51}$$

with $L = 5$ and $M_u = 0.1$.

Discrete planning. The last technique is an adaptation of the method proposed in (Kim and Eustice, 2013). The approach (Kim and Eustice, 2013), called PDN (Perception-Driven Navigation), selects a set of waypoints, that corresponds to potential loop closure locations. PDN operates over a pose-graph and calculates, for each waypoint, a path from the current position to the waypoint, using a global $A^\star$ algorithm over the nodes in the pose-graph. In case the distance between adjacent nodes is too great, the method interpolates between the poses to yield finer segments. The $A^\star$ algorithm uses a heuristic weighting, in

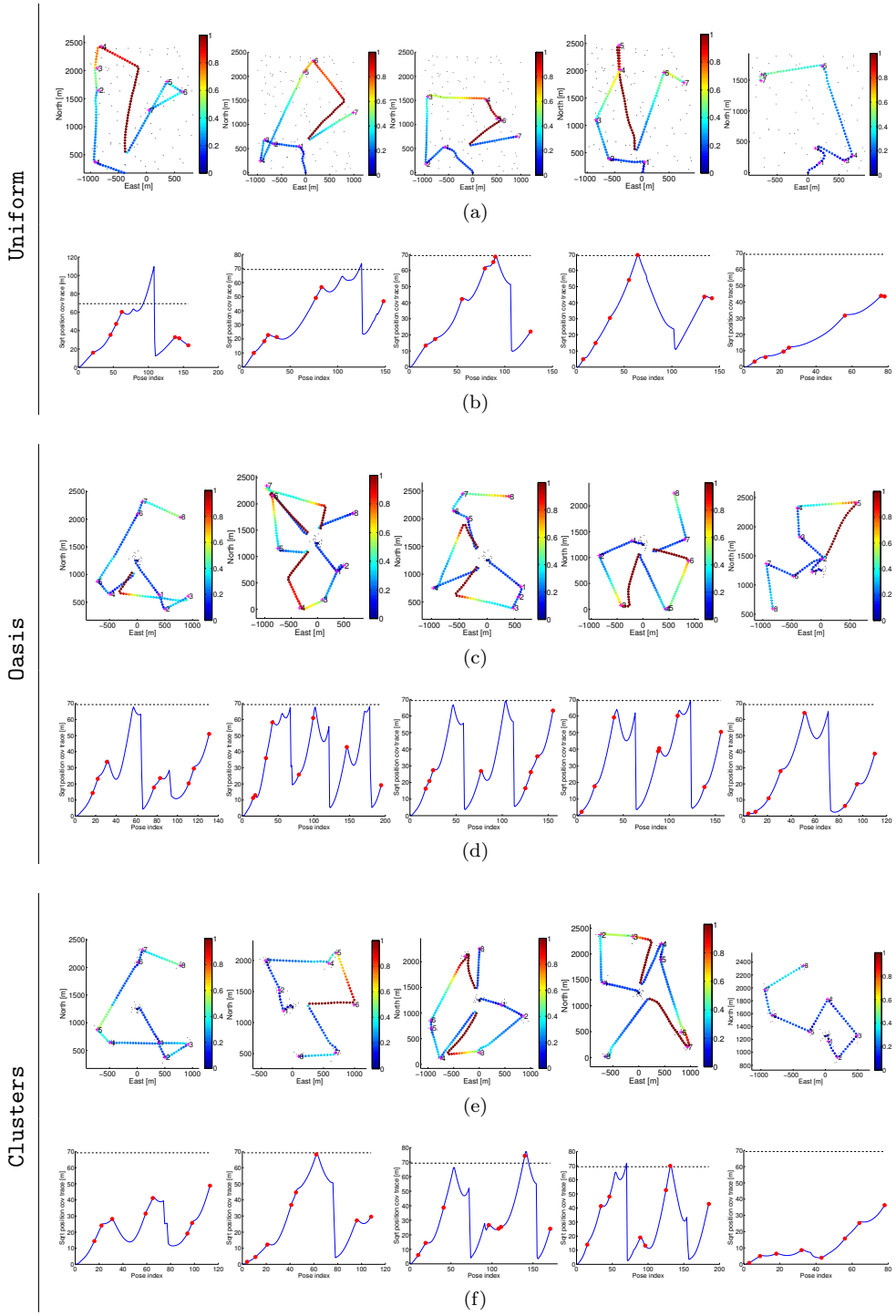


Figure 13: Trajectories and uncertainty for 5 random goal configurations (columns) and UNIFORM, OASIS, CLUSTERS and scenarios (rows).

which one assigns a *saliency* value in $[0, 1]$ to each node in the grid. The heuristic rewards the robot for visiting nodes with large saliency; intuitively, nodes with large saliency are the ones in which the robot is more likely to successfully re-observe landmarks. Saliency is also used to model the probability of making camera observations, i.e. the likelihood of making measurements along a path. Then, the utility of each path is computed by propagating the belief along this path and computing a cost associated to this realization of this belief. This cost balances the two contrasting objectives of minimizing uncertainty and exploring new areas in the scenario. We apply some modifications to the approach (Kim and Eustice, 2013), in order to have a fair comparison with our technique. First, (Kim and Eustice, 2013) considers a pose-only estimation framework. Instead, for a fair comparison, we adopt the same estimation framework of Section 3.2.1, in which the state may include both robot poses and landmark locations. We discretize the state space (e.g. 3D positions) and consider a regular grid with resolution representing the above-mentioned interpolation. In order to identify loop closures opportunities, we cluster the observed landmarks and we use each cluster center as potential waypoint in PDN. Second, (Kim and Eustice, 2013) assumes that a reference trajectory is preplanned and the robot only has to decide when revisiting a past pose or continue along the given trajectory. Since we do not have a preplanned trajectory to follow (we rather have goals), we include the current goal in the set of waypoints, such that the robot can autonomously decide between visiting previously observed landmarks or moving towards the goal. Finally, instead of using the cost specified in (Kim and Eustice, 2013), we use the objective function (41), such that PDN and our approach attempt to minimize the same function. In the following, we refer to our adaptation of (Kim and Eustice, 2013) with the name "Discrete" planning; this name stresses the fact that, as in most related works, this technique discretizes the state space and selects the trajectory among a finite number of candidate paths.

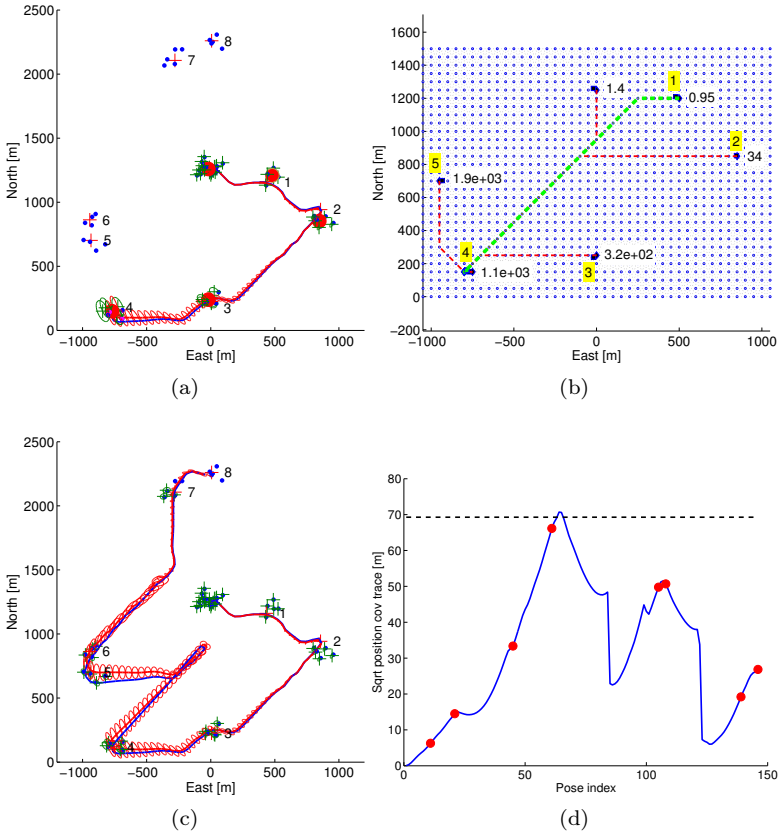


Figure 14: Discrete approach: (a) Current robot trajectory, with goal and landmark clusters centers (red filled circles). (b) Grid world used by the $A^\star$ algorithm and paths to each waypoint (indicated by a number with yellow background). Waypoints include landmark cluster centers and next goal position. Selected path (with best objective) is shown in green. (c) Complete trajectory and (d) square root of covariance trace.

Fig. 14 elaborates on the main components of the Discrete approach, considering a typical run in the CLUSTERS scenario: the resulting robot trajectory and the square root of the position covariance trace are shown in Figs. 14c-14d; Fig. 14a pictures robot trajectory and the considered waypoints right after reaching goal 4. These include the centers of landmark clusters, denoted by large red filled circles, and the position of the next goal (goal 5). Fig. 14b shows the discretization grid, the calculated trajectories and the associated costs to each of these waypoints. We use a 50 meters grid resolution for the Discrete method in all experiments. The waypoint with the best (smallest) objective is chosen (green trajectory). The weights used in the objective function are analogous to the GBS approach. As in all other approaches, we use an MPC philosophy: we

compute a plan at each time step and apply a single step of this plan.

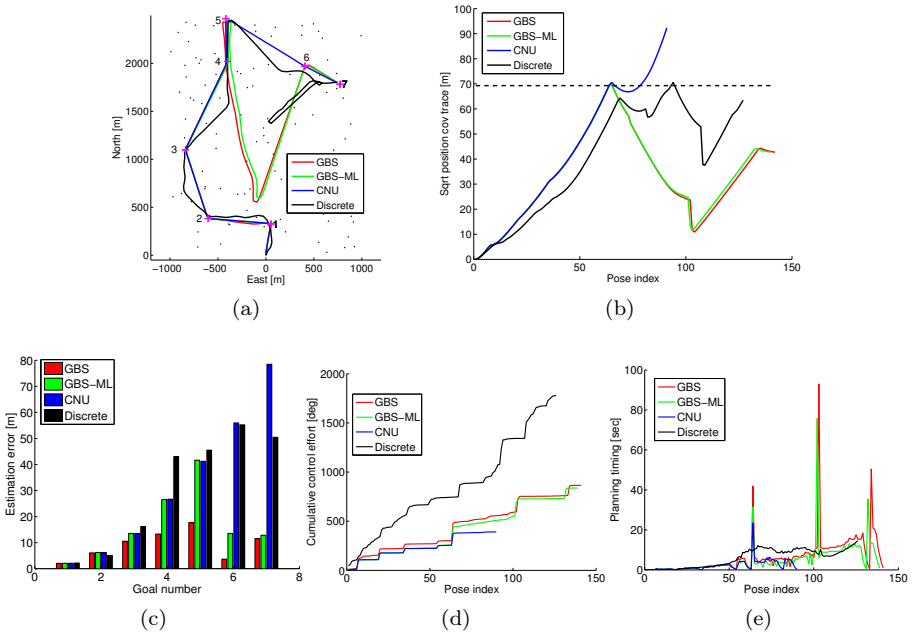


Figure 15: Comparison between different planning approaches for UNIFORM scenario. (a) Trajectories; (b) square root of position covariance trace; (c) Miss distance; (d) Cumulative control effort; (e) Planning time.

7.2.1 Results

We test the compared techniques in the UNIFORM, OASIS, CLUSTERS scenarios.

We first discuss in detail the UNIFORM scenario. Fig. 15a shows the trajectories produced by the four approaches, while in Fig. 15b, we report, for each approach, the trace of the corresponding covariance. CNU approach disregards uncertainty and only rewards small distance to the goal and small control efforts, see Eq. (51). Accordingly, at each time step, CNU leads the robot directly to the goal. The drawback, in this case, is that the CNU approach does not attempt to bound uncertainty, and the corresponding covariance trace grows unbounded, as shown in Fig.15b. On the other hand, GBS and GBS-ML aim at keeping uncertainty below the bound $\beta \approx 69^2$ m^2. After reaching goal 5, the trace of the covariance is close to the specified upper bound, therefore, both techniques prefer to revisit landmarks instead of proceeding to goal 6. This leads to the uncertainty reduction in the green and red lines of Fig. 15b. In the considered scenario, the GBS approach produces slightly different plans than the GBS-ML technique. The term in Eq. (46) that appears when relaxing the maximum likelihood assumption, rewards uncertainty reduction; for this reason,

in our example, the GBS tends to perform loop closure before GBS-ML. Also the discrete approach is able to account for robot uncertainty and produces a plan that satisfies the soft uncertainty bound.

The capability of imposing an upper bound on the uncertainty influences the capability of having small estimation errors. In Fig. 15c we show the miss distances for the compared approaches. The GBS approach is able to plan a trajectory that reaches the 7 goals (x-axis in the histogram) with small errors.

Regarding the comparison between the proposed approach and the Discrete approach, we observe from Fig. 15a that the paths produced by the Discrete planner are less natural and include many redundant changes in robot direction. This behavior can be easily understood from Fig. 14b: since $A^\star$ works on a grid world, the robot can move only to the 8 neighboring cells, and cannot perform fine adjustments of robot position and orientation. Accordingly, as in Figs. 1 and 14c, instead of producing straight trajectories to the goals, the planner produces segments with orientations $k\frac{\pi}{4}$, with $k = 0, \ldots, 7$. This leads to many discontinuities in the trajectory, that are only caused by the fact that the approach relies on discretization. On the other hand, the approach proposed in this paper works in a continuous domain and produces smoother and more natural trajectories. This qualitative observation is further confirmed by Fig. 15d, that shows the cumulative control usage for the different techniques. The control effort is similar for the techniques working in the continuous domain (GBS, GBS-ML, CNU), while it is remarkably higher for the discrete planner.

Fig. 15e reports the computation time spent in planning for the different approaches. The time is lower for the CNU approach, and this is expected, as this approach uses a simplified model. On the other hand, the CPU time is similar for the remaining techniques, with the only difference that the proposed approach has "peaks" in the CPU time, corresponding to larger number of iterations in the gradient method. These peaks are associated with significant changes in the heading angle, as discussed in Section 7.1.3.

Figures 16 and 17 provide a comparison of the 4 approaches in Oasis and Clusters scenarios. The results support the above discussion and further confirm our findings. GBS is able to correctly impose the uncertainty bound. Moreover, it produces more natural trajectories, when compared to a Discrete approach, and requires a smaller control effort. In these two scenarios, the computational time is smaller for the Discrete approach (Figs. 16e and 17e). This is due to the fact that in these scenarios the landmarks are clustered by construction, hence the number of waypoints for the discrete approach is small, which, in turn, implies a smaller number of paths to be computed and evaluated.

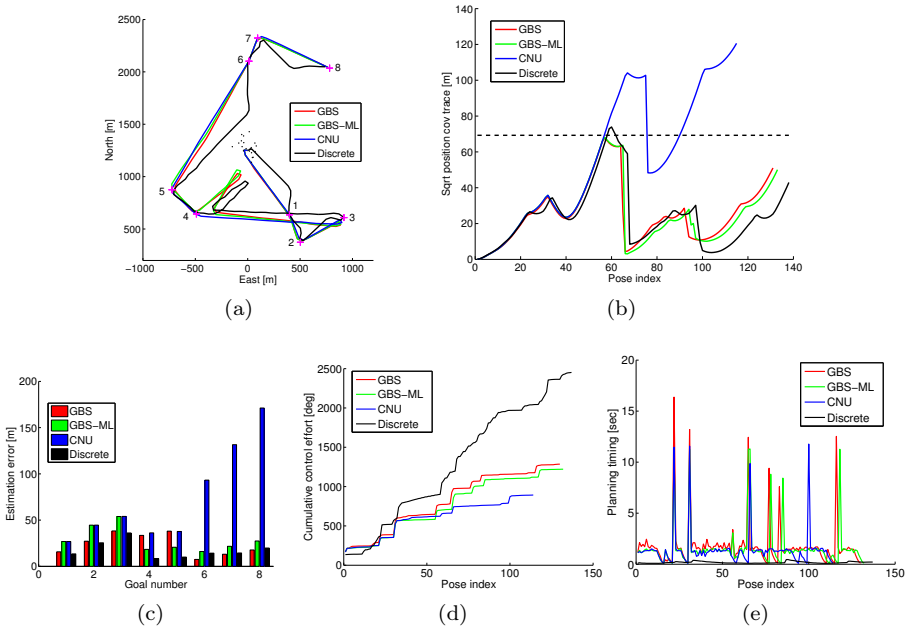


Figure 16: Comparison between different planning approaches for Oasis scenario. (a) Trajectories; (b) square root of position covariance trace; (c) Miss distance; (d) Cumulative control effort; (e) Planning time.

In order to further examine the performance of the proposed approach and compare it against GBS-ML, CNU and Discrete, we consider different random goal configurations in all the three scenarios (Uniform, Oasis, Clusters). The randomly generated goals are shown in Fig. 13. We perform statistics over these runs and we summarize the results in Table 1. The performance metrics are the ones presented at the beginning of Section 7: trajectory length, estimation errors, cumulative control usage, and planning time. Mean (μ) and standard deviation (σ) for each of these metrics are presented, calculated by analyzing the entire trajectories in all runs. Table 1 confirms the following findings:

- Trajectory length and estimation errors are typically smaller for the GBS planning, compared to the Discrete approach.
- Control usage is significantly smaller for GBS planning compared to the Discrete approach.
- Planning time is higher for GBS planning, compared to Discrete planning, in Oasis and Clusters scenarios (that contain only a few clustered landmarks). Planning time is smaller for GBS in the presence of a large number of uniformly distributed landmarks.

- Trajectory length and control effort are small for CNU approach. However, this comes at the expense of much higher estimation errors and covariances (not shown in the table, see Figs. 15b, 16b and 17b).
- Overall performance of GBS and GBS-ML approaches is comparable, and they can both assure satisfaction of the uncertainty bound and efficient goal accomplishment.

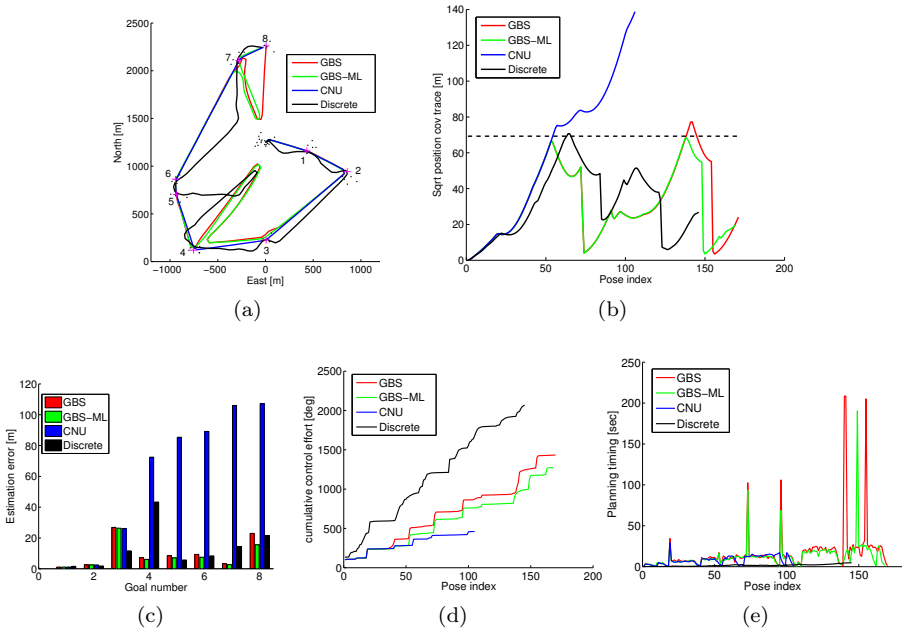


Figure 17: Comparison between different planning approaches for CLUSTERS scenario. (a) Trajectories; (b) square root of position covariance trace; (c) Miss distance; (d) Cumulative control effort; (e) Planning time.

8 Discussion of Possible Extensions

The experimental evaluation, whose results were presented in Section 7, suggests several avenues for future work. In this section we discuss some desirable extensions of the proposed approach that can enhance its applicability in challenging scenarios.

8.1 Improving Effectiveness of Weight Matrices Selection

In Section 6.1 we anticipated that the choice of the weight matrices can be scenario-dependent and hard to manage in general. We also remarked that

Scenario & Metric		**GBS**		GBS-ML		CNU		Discrete	
		μ	σ	μ	σ	μ	σ	μ	σ
Uniform	Traj. Length	**109**	**39**	108	39	89	17	155	94
	Est. Err. [m]	**16.7**	**10.8**	15.8	9.8	18.6	15.6	25.8	15.0
	Control [deg]	**758**	**138**	678	142	507	103	2067	993
	Timing [sec]	**6.7**	**14.6**	5.5	11.2	2.8	4.8	10.7	9.3
Oasis	Traj. Length	**149**	**31**	148	31	113	8	144	38
	Est. Err. [m]	**17.2**	**13.2**	20.7	18.1	79.7	84.7	15.2	11.9
	Control [deg]	**1503**	**412**	1535	473	856	187	2381	689
	Timing [sec]	**1.8**	**2.9**	1.7	2.7	1.6	2.9	0.19	0.1
Clusters	Traj. Length	**131**	**45**	129	43	103	20	150	62
	Est. Err. [m]	**9.8**	**8.7**	9.5	9.4	57.4	82.6	11.5	9.2
	Control [deg]	**1124**	**307**	952	250	617	120	2276	773
	Timing [sec]	**11.5**	**18.4**	9.8	13.5	8.2	8.7	2.6	2.4

Table 1: Performance evaluation in Uniform, Oasis, Clusters with randomly chosen goals (See Fig. 13). Performance metrics: trajectory length, position estimation errors, cumulative control, planning time. Results are shown in terms of mean (μ) and standard deviation (σ).

choosing large values of M_u can create undesirable behavior (Fig. 6b), as the penalty on control usage interferes with the other terms in the objective function. Ideally, one would like to select weights to specify the "importance" of each term (uncertainty reduction, goal attainment, control usage). In this section we provide additional insight as to why this selection is non-trivial and also suggest how to extend our approach to improve its effectiveness.

To provide more insight we discuss a simplified example where we neglect uncertainty and consider only two terms to balance in the objective function, namely control usage and goal attainment. Under these assumptions, Eq. (41), for a single look-ahead step, becomes:

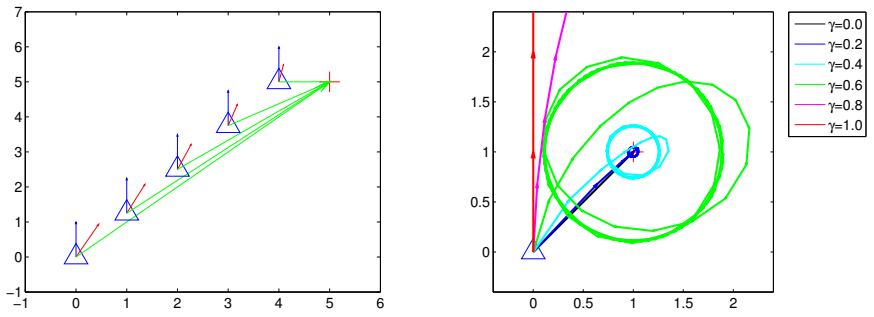
$$J_k\left(u_k\right) \doteq \left\|\zeta\left(u_k\right)\right\|_{M_u}^2 + \left\|x_{k+1} - X^G\right\|_{M_x}^2 \tag{52}$$

where we reward next robot position x_{k+1} to be close to the goal X^G. To make the problem even simpler, we assume a linear motion model: $x_{k+1} = x_k + u_k$, where $x_k \in \mathbb{R}^3$ is the robot position at time k and $u_k \in \mathbb{R}^3$ is the control action we need to plan. We also assume $\zeta\left(u_k\right) \doteq u_k - \bar{u}_k$, meaning that we penalize differences between the planned control u_k and a given nominal control $\bar{u}$. For instance, we can set $\bar{u}_k = V\frac{u_{k-1}}{\|u_{k-1}\|}$, meaning that the nominal control imposes a constant speed V and penalizes changes in the direction $\frac{u_{k-1}}{\|u_{k-1}\|}$. In this example we are neglecting uncertainty, hence the state is predicted exactly by the model $x_{k+1} = x_k + u_k$. Finally, we assume the weight matrices M_u and M_x to be in the form $M_u = \gamma I_3$ and $M_x = (1-\gamma) I_3$, where $\gamma \in [0,1]$, and I_3 is a 3×3 identity matrix. Under these assumptions, (52) simplifies to:

$$J_k\left(u_k\right) \doteq \gamma\left\|u_k - \bar{u}\right\|^2 + (1-\gamma)\left\|u_k + x_k - X^G\right\|^2. \tag{53}$$

In this simple case, the cost is quadratic in u_k, and it is possible to compute the optimal control $u_k^\star$ in closed form:

$$u_k^\star = \gamma \bar{u} + (1-\gamma)\left(X^G - x_k\right). \tag{54}$$



(a) (b)

Figure 18: (a) Optimal control $u_k^\star$ (red arrow) for different robot positions (blue $\triangle$). The control $u_k^\star$ at a given position is a linear combination of the nominal control $\bar{u}_k$ (blue arrow) and the vector $X^G - x_k$ (green arrow). (b) Planned trajectories for different values of γ; robot starts at the origin and has to reach goal position (1,1).

Fig. 18a shows the vector $u_k^\star$ (in red) for different robot positions x_k and for a fixed $\gamma = 0.7$. Ideally, we want to use γ to impose a given trade-off between goal attainment and fidelity to the nominal control $\bar{u}_k$; in particular, we would like the following behavior: for a fixed $\gamma \to 1$ the control is closer to the nominal one, while for small γ we prioritize reaching the goal. However, one may notice from (54) that when the robot is far away from the goal, the term $X^G - x_k$ is dominating, even for large γ. Therefore, our ideal behavior cannot be assured in general. This is confirmed by Fig. 18a, that shows that when the robot is far from the goal, the control vector $u_k^\star$ (in red) gets closer to the vector $X^G - x_k$ (in green). The behavior is undesirable, as when we are far away from the goal, we apply very strong control actions (arbitrarily neglecting the nominal control $\bar{u}_k$, in blue in Fig. 18a), while, close to the goal, we apply weak control and we mainly follow the nominal control $\bar{u}_k$. The dependence on the distance to the goal $X^G - x_k$ makes it hard to establish a good policy for the weight, and this is a consequence of the use of a quadratic cost, in which the "attraction" towards the goal grows quadratically with the distance.

In Fig. 18b we simulate the planner (54) for different values of γ, assuming the robot starts at the origin and the goal is located at $(1, 1)$. This figure further remarks the importance of the choice of the weights. For small values of γ the robot "orbits" around the goal and is unable to reach it, as we observed in our nonlinear planning problem of Fig. 6b. This is again undesirable: the reason

is again that the control action is position dependent and the attraction to the goal becomes weaker when the robot is closer to the goal itself.

Note that in the considered example it may seem easy to balance the terms. However, in the general case of Eq. (41) is nontrivial to balance the cost components and the issue discussed in this section may manifest in other ways. For instance, when the robot is far from the goal, the cost component related to goal attainment dominates, and the robot may prefer to go towards the goal, even it this leads to violation of the uncertainty bounds. In this paper, our solution to cope with this issue was to make the weights a function of robot uncertainty via the parameter α_k (Section 6.1), instead of fixing them. This is essentially a way to prioritize uncertainty reduction, when robot uncertainty is close to the specified upper bound β. This choice has few undesirable drawbacks; for instance, when α_k is small, the cost of reaching a far goal is still dominating, and this may lead the robot to miss loop closing opportunities.

In future work we plan to explore two solutions to facilitate the choice of the weights. The first solution is a practical one, and consists in the use of intermediate "waypoints" on the way from the robot to the goal. Setting the waypoints at fixed distances allows to make the term $X^G - x_k$ constant, such that a fixed choice of γ can regulate the two terms in Eq. (54).

A second solution consists in the use of the ℓ_1 norm instead of the Euclidean norm for distances. This removes the dependence of the control action on the distance to the goal, but presents other challenges, e.g., non-differentiability at the origin, and difficulty to compute the expectation in Eq. (40).

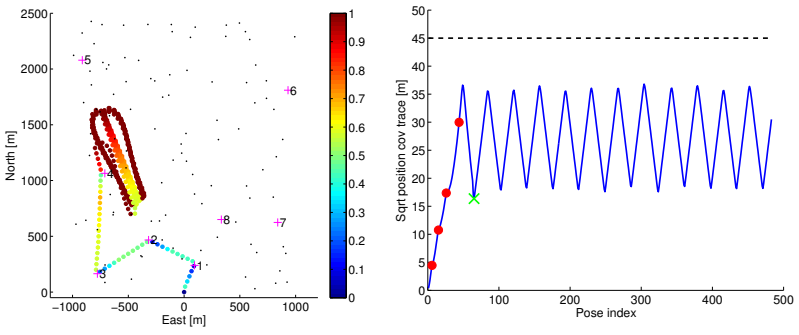
8.2 Coping with Loopy Solutions

A second way to improve our approach is to exploit domain-specific knowledge to inform the planner about the feasibility of a given problem instance.

Let us discuss this point using the example in Fig. 19a. In this case we impose a stricter uncertainty bound $\beta = 40^2\text{m}^2$. The robot reaches goal 4 and, while moving towards goal 5, decides to close the loop, and come back to observe known landmarks. Using our heuristic (Section 6.1.2), the planner requires that the uncertainty is sufficiently small before heading back to goal 5. Therefore, the robot keeps revisiting known areas until the uncertainty reaches the point labeled with $\times$ in Fig. 19b. At that point the uncertainty is sufficiently small, and the robots heads back to goal 5. However, the *uncertainty budget* is not sufficient to reach the goal, and the robot gets stuck in the loopy behavior of Fig. 19a; the uncertainty keeps oscillating, as shown in Fig. 19b.

The very same problem may occur, even if we ask the robot to re-establish an uncertainty close to zero, before moving to the goal. There are cases in which the goal is simply out-of-reach, given the initial uncertainty budget (which is a function of measurement and motion noise). The impossibility to reach the goal within admissible uncertainty is an intrinsic limitation of the problem instance.

A practical solution to cope with this problem is the following. Once the loopy condition of Fig. 19a is detected, the planner can simply relax the uncertainty bound, to enable the robot to reach goal 5. In the results presented



(a) (b)

Figure 19: (a) Example in which the planner fails and the robot keeps oscillating between the competing behaviors of uncertainty reduction and goal attainment. (b) Square root of the trace of position covariance; the green $\times$ indicates that the planner induces the robot to head back to the goal.

in this paper we did not need to implement this heuristic, while it may be required when the uncertainty bounds are very strict. Future work includes to investigate in more details the concept of uncertainty budget, and how to use it beforehand, e.g., to predict which goal is reachable or dynamically adjusting uncertainty bounds depending on the scenario.

8.3 Enforcing Hard Constraints

So far we presented an approach that allows imposing uncertainty bounds in the form of soft constraints. This means that the planner allows (but penalizes) violations of such bounds. In many applications, one may be interested in imposing hard constraints on uncertainty, meaning that the plan has to *guarantee* that uncertainty is below the bound. In this section we briefly discuss how to extend the proposed formulation to cope with hard constraints.

Let us start from the cost (41), and let us move the covariance trace from the objective function, and use it to constrain the optimization problem:

$$\begin{aligned} \min_{u_{k:k+L-1}} \quad & J_k\left(u_{k:k+L-1}\right) \\ s.t. \quad & tr\left(M_\Sigma I_{k+l}^{-1} M_\Sigma^T\right) \leq \beta \quad l = 1, \ldots, L \end{aligned} \tag{55}$$

with:

$$\begin{aligned} J_k\left(u_{k:k+L-1}\right) \doteq \textstyle\sum_{l=0}^{L-1} \left\|\zeta\left(u_{k+l}\right)\right\|_{M_u}^2 + \left\|E_{k+L}^G \bar{X}_{k+L} - X^G\right\|_{M_x}^2 \\ + tr\left(Q_{k+L}\left(H_{k+L} \bar{I}_{k+L}^{-1} H_{k+L}^T + \Omega_v^{-1}\right)\right). \end{aligned} \tag{56}$$

In order to frame the constrained optimization problem (56) intro an unconstrained one, we use a *log barrier* method. In the log barrier method, for each

constraint $tr\left(M_\Sigma I_{k+l}^{-1} M_\Sigma^T\right) \leq \beta$, we add a term in the form $-\log\left(tr\left(M_\Sigma I_{k+l}^{-1} M_\Sigma^T\right) - \beta\right)$ in the objective function:

$$J_k\left(u_{k:k+L-1}\right) \doteq \sum_{l=0}^{L-1} \left\|\zeta\left(u_{k+l}\right)\right\|_{M_u}^2 - \sum_{l=0}^{L} \log\left(tr\left(M_\Sigma I_{k+l}^{-1} M_\Sigma^T\right) - \beta\right) + \left\|E_{k+L}^G \bar{X}_{k+L} - X^G\right\|_{M_x}^2 + tr\left(Q_{k+L}\left(H_{k+L} \bar{I}_{k+L}^{-1} H_{k+L}^T + \Omega_v^{-1}\right)\right). \quad (57)$$

Intuitively, when the trace $tr\left(M_\Sigma I_{k+l}^{-1} M_\Sigma^T\right)$ at the lth look-ahead step approaches β, the argument of the logarithm tends to zero, and then the negative logarithm tends to infinity. Therefore, all actions violating the bound β appear to have infinite cost. While this solution may seem very easy and appealing, it still has some drawbacks. The main issue regards the fact that, as discussed in Section 8.2, one cannot guarantee the existence of a path from the robot position to the goal, such that the uncertainty constraint is satisfied. Therefore, a formulation with hard constraints can easily lead to infeasible problem instances. In the proposed GBS approach, instead, we penalize violations of the bound, but we still manage cases in which satisfying this bound is not possible.

Future work includes a deeper investigation of the theoretical guarantees related to the proposed approach (e.g., guaranteed uncertainty bounds). To the best of our knowledge, the only approach in this direction is the one presented in (Carlone and Lyons, 2014). However, in (Carlone and Lyons, 2014) the authors consider a linear model for the system (robot motion and measurements), and they impose that the uncertainty bound is only satisfied at the *end* of the time horizon, while enabling bound violation at intermediate look-ahead steps.

8.4 Other Extensions

In this work we did not discuss obstacle avoidance during planning. Modeling obstacles is possible within our framework, and we refer the interested reader to (Van Den Berg et al., 2012) and the references therein. Future work also includes investigation of the relation to the mixed-integer formulation of (Carlone and Lyons, 2014), in which the planner is *guaranteed* to produce collision-free trajectories.

From a practical standpoint, we feel that more work needs to be done to make the planning approach more efficient. Also on this front, many solutions are possible, ranging from more efficient implementations of the gradient method, the design of specific objective functions in which analytic differentiation is viable, to the investigation of nonlinear optimization methods with faster convergence.

9 Conclusion

This work investigates the problem of planning under uncertainty, and addresses several limitations of state-of- the-art techniques, namely (i) state or control discretization, (ii) assumption of maximum likelihood observations, and (iii) assumption of prior knowledge about the environment in which the robot operates.

We propose a planning approach that operates in a continuous domain. The approach is based on a dual-layer architecture, with an *inner inference layer*, which is in charge of predicting the belief on robot and world states for a given control action, and *outer decisional layer*, which has to compute an optimal control strategy using the predictions of the inner layer. In the inner inference layer, we treat future measurements as random variables, hence avoiding the assumption of maximum likelihood observations. We also include random binary variables to model the fact that it may be not known in advance whether a measurement is acquired or not. Our approach works in the *generalized belief space*, which, besides robot and world state, encodes a joint probability distributions over future measurements and latent binary variables. We test our approach in realistic simulation scenarios; experimental results show that the approach is able to deal with the concurrent tasks of exploring an unknown environment and keeping the uncertainty bounded. Moreover, the approach produces smoother and more natural trajectories with respect to related approaches that rely on discretization. We further use the results to discuss current limitations of our approach and elaborate on possible avenues for future work.

Funding

This work was partially funded by the ARL Micro Autonomous Systems and Technology (MAST) program under contract No. W911NF-08-2-0004 "Distributed Inference: Scalable Online Robust Decentralized Inference for Active Exploration" and by the National Science Foundation Award 11115678 "RI: Small: Ultra-Sparsifiers for Fast and Scalable Mapping and 3D Reconstruction on Mobile Robots". The views expressed in this work have not been endorsed by the sponsors.

Appendix A: Index to Multimedia Extension

The multimedia extension page is found at http://www.ijrr.org.

Table of Multimedia Extension

Extension	Media type	Description
1	Video	Application of planning in GBS to autonomous navigation in unknown environments

Appendix B: Derivation of Eq. (41)

In this appendix we complete the derivation of Eq. (41) from Eq. (40), deriving the expectation $\underset{Z_{k:k+l}}{\mathbb{E}}\left[\left\|E^G_{k+l}X^\star_{k+l} - X^G\right\|^2_{M_x}\right]$. We present the derivation for the general case of the lth look-ahead step, which is valid in particular for $l = L$ as it appears in Eq. (40).

We first substitute $X^{\star}_{k+l}$, as given in Eq. (37), into the expectation:

$$\underset{Z_{k:k+l}}{\mathbb{E}}\left[\left\|E^G_{k+l}X^{\star}_{k+l}-X^G\right\|^2_{M_x}\right]=\underset{Z_{k:k+l}}{\mathbb{E}}\left[\left\|E^G_{k+l}\bar{X}_{k+l}+E^G_{k+l}I^{-1}_{k+l}\mathcal{A}^T_{k+l}\breve{b}_{k+l}-X^G\right\|^2_{M_x}\right],$$

where we exploited that $\left(\mathcal{A}^T_{k+l}\mathcal{A}_{k+l}\right)^{-1}\mathcal{A}^T_{k+l}\breve{b}_{k+l}=I^{-1}_{k+l}\mathcal{A}^T_{k+l}\breve{b}_{k+l}$, as per Eq. (44). Now we notice that choosing the nominal state $\bar{X}_{k+l}$ as in Eq. (27) annihilates $\breve{b}^f_{k+l}$ in Eq. (32), and implies $\mathcal{A}^T_{k+l}\breve{b}_{k+l}=\mathcal{H}^T_{k+l}\breve{b}^h_{k+l}$; then our expectation becomes

$$\underset{Z_{k:k+l}}{\mathbb{E}}\left[\left\|E^G_{k+l}X^{\star}_{k+l}-X^G\right\|^2_{M_x}\right]=\underset{Z_{k:k+l}}{\mathbb{E}}\left[\left\|E^G_{k+l}\bar{X}_{k+l}+E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{b}^h_{k+l}-X^G\right\|^2_{M_x}\right].$$

Observe that both $\mathcal{H}^T_{k+l}$ and $\breve{b}^h_{k+l}$ are whitened by the scaled measurement information matrices $\bar{\Omega}^{ij}_v$ (defined in Eq. (25)). Developing the squared norm we rewrite the expectation as

$$\begin{aligned}\underset{Z_{k:k+l}}{\mathbb{E}}\left[\left\|E^G_{k+l}\bar{X}_{k+l}+E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{b}^h_{k+l}-X^G\right\|^2_{M_x}\right]&=\underset{Z_{k:k+l}}{\mathbb{E}}\Big[\left\|E^G_{k+l}\bar{X}_{k+l}-X^G\right\|^2_{M_x}\\&+\left\|E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{b}^h_{k+l}\right\|^2_{M_x}-2\left(E^G_{k+l}\bar{X}_{k+l}-X^G\right)^T M_x\left(E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{b}^h_{k+l}\right)\Big]\end{aligned}\quad(58)$$

The vector $\breve{b}^h_{k+l}$ includes the whitened innovation terms $\left(\bar{\Omega}^{i,j}_v\right)^{\frac{1}{2}}b^h_{i,j}\left(z_{k+i,j}\right)$, $i\in[1,\dots,l]$ and $j\in[1,n_i]$, see Eqs. (30) and (31). Therefore, we can write

$$\breve{b}^h_{k+l}=\breve{\Omega}^{\frac{1}{2}}_v b^h_{k+l},$$

where b^h_{k+l} is constructed by stacking the innovations $b^h_{i,j}\left(z_{k+i,j}\right)$.

Since b^h_{k+l} is the only term depending on the observations $Z_{k:k+l}$, we rewrite Eq. (58) as:

$$\begin{aligned}&\left\|E^G_{k+l}\bar{X}_{k+l}-X^G\right\|^2_{M_x}+\underset{Z_{k:k+l}}{\mathbb{E}}\left[\left\|E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{\Omega}^{\frac{1}{2}}_v b^h_{k+l}\right\|^2_{M_x}\right]\\&-2\left(E^G_{k+l}\bar{X}_{k+l}-X^G\right)^T M_x\left(E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{\Omega}^{\frac{1}{2}}_v\underset{Z_{k:k+l}}{\mathbb{E}}\left[b^h_{k+l}\right]\right).\end{aligned}\quad(59)$$

In order to evaluate the expectation $\underset{Z_{k:k+l}}{\mathbb{E}}\left[b^h_{k+l}\right]$, we write explicitly $b^h_{i,j}\left(z_{k+i,j}\right)$ as in Eq. (30):

$$b^h_{i,j}\left(z_{k+i,j}\right)\doteq z_{k+i,j}-h\left(\bar{X}^o_{k+i,j}\right)=h\left(X^o_{k+i,j}\right)-h\left(\bar{X}^o_{k+i,j}\right)+v_{i,j}\quad,\quad v_{i,j}\sim N\left(0,\Omega^{ij}_v\right)$$

where we also substituted the measurement $z_{k+i,j}$ using Eq. (6). We notice that $X^o_{k+i,j}$ is the true value of the state (unknown in practice) while $\bar{X}^o_{k+i,j}$ is the nominal belief, computed as in Eq. (27). We write $\bar{X}^o_{k+i,j}=X^o_{k+i,j}+\tilde{X}^o_{k+i,j}$ where $\tilde{X}^o_{k+i,j}$ is the mismatch between the estimate $\bar{X}^o_{k+i,j}$ and the true value $X^o_{k+i,j}$. Now we linearize the measurement model around the nominal value $\bar{X}^o_{k+i,j}$, obtaining

$$b^h_{i,j}\left(z_{k+i,j}\right)\approx h\left(\bar{X}^o_{k+i,j}\right)+H_{i,j}\tilde{X}^o_{k+i,j}-h\left(X^o_{k+i,j}\right)+v_{i,j}=H_{i,j}\tilde{X}^o_{k+i,j}+v_{i,j}\quad(60)$$

where $H_{i,j} \doteq \nabla_x h$ is the Jacobian of the measurement model $h(\cdot)$. Considering all the measurements, and the overall joint state at the lth look-ahead step, X_{k+l}, we can write the following compact expression:

$$b^h_{k+l} = H_{k+l}\tilde{X}_{k+l} + \Upsilon \tag{61}$$

where Υ is a vector including all the noise terms $v_{i,j}$. By construction, $\Upsilon \sim N(0, \Omega_v)$, where Ω_v is a block diagonal matrix containing Ω^{ij}_v on the diagonal. Recall that the nominal state $\bar{X}_{k+i}$ coincides with the prior estimates (see Eq. (27)), i.e., the estimate of the state given the controls, but before including the measurements. Therefore, $\tilde{X}_{k+i} \sim N(0, \bar{I}_{k+l})$, where $\bar{I}_{k+l}$ is the information matrix of the prior estimate given in Eq. (44).

From Eq. (61) it follows that

$$\begin{cases} \mathop{\mathbb{E}}_{Z_{k:k+l}}\left[b^h_{k+l}\right] = 0 & \text{(mean)} \\ \mathop{\mathbb{E}}_{Z_{k:k+l}}\left[b^h_{k+l}\left(b^h_{k+l}\right)^T\right] = H\bar{I}^{-1}_{k+l}H^T + \Omega^{-1}_v & \text{(covariance)} \end{cases} \tag{62}$$

Using the first line in Eq. (62), the last term in Eq. (59) disappears:

$$\left\|E^G_{k+l}\bar{X}_{k+l} - X^G\right\|^2_{M_x} + \mathop{\mathbb{E}}_{Z_{k:k+l}}\left[\left\|E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{\Omega}^{\frac{1}{2}}_v b^h_{k+l}\right\|^2_{M_x}\right] \tag{63}$$

Now we simplify the second term in Eq. (63) by writing in explicit form the Mahalanobis norm:

$$\left\|E^G_{k+l}\bar{X}_{k+l} - X^G\right\|^2_{M_x} + \mathop{\mathbb{E}}_{Z_{k:k+l}}\left[\left(E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{\Omega}^{\frac{1}{2}}_v b^h_{k+l}\right)^T M_x \left(E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{\Omega}^{\frac{1}{2}}_v b^h_{k+l}\right)\right] =$$
$$\left\|E^G_{k+l}\bar{X}_{k+l} - X^G\right\|^2_{M_x} + \mathop{\mathbb{E}}_{Z_{k:k+l}}\left[\left(b^h_{k+l}\right)^T Q_{k+l}\left(b^h_{k+l}\right)\right]$$

where

$$Q_{k+l} = \left(E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{\Omega}^{\frac{1}{2}}_v\right)^T M_x \left(E^G_{k+l}I^{-1}_{k+l}\mathcal{H}^T_{k+l}\breve{\Omega}^{\frac{1}{2}}_v\right).$$

Now we use the fact that, for a random vector y and a given matrix Q, it holds $\mathop{\mathbb{E}}_{y}\left[y^T Q y\right] = \mathop{\mathbb{E}}_{y}[y]^T Q \mathop{\mathbb{E}}_{y}[y] + tr(Q\Sigma_y)$, where Σ_y is the covariance of y. Using this property and recalling Eq. (62) we conclude:

$$\mathop{\mathbb{E}}_{Z_{k:k+l}}\left[\left\|E^G_{k+l}X^\star_{k+l} - X^G\right\|^2_{M_x}\right] = \left\|E^G_{k+l}\bar{X}_{k+l} - X^G\right\|^2_{M_x} + tr\left(Q_{k+l}\left(H_{k+l}\bar{I}^{-1}_{k+l}H^T_{k+l} + \Omega^{-1}_v\right)\right) \tag{64}$$

where $\bar{I}_{k+l}$ is the information matrix of the prior estimate.

References

Bai, H., David, H., and Lee, W. (2013). Integrated perception and planning in the continuous space: A POMDP approach. In *Robotics: Science and Systems (RSS)*.

Binney, J. and Sukhatme, G. S. (2012). Branch and bound for informative path planning. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 2147–2154.

Blanco, J., Fernandez-Madrigal, J., and Gonzalez, J. (2008). A novel measure of uncertainty for mobile robot SLAM with Rao-Blackwellized particle filters. *Intl. J. of Robotics Research*, 27(1):73–89.

Bryson, M. and Sukkarieh, S. (2008). Observability analysis and active control for airborne SLAM. *IEEE Trans. Aerosp. Electron. Syst.*, 44:261–280.

Carlone, L. and Lyons, D. (2014). Uncertainty-constrained robot exploration: a mixed-integer linear programming approach. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*.

Carrillo, H., Reid, I., and Castellanos, J. (2012). On the comparison of uncertainty criteria for active SLAM. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 2080–2087.

Du, J., Carlone, L., Ng, M. K., Bona, B., and Indri, M. (2011). A comparative study on active SLAM and autonomous exploration with particle filters. In *IEEE/ASME Intl. Conf. on Advanced Intelligent Mechatronics (AIM)*, pages 916–923.

Erez, T. and Smart, W. D. (2010). A scalable method for solving high-dimensional continuous pomdps using local approximation. In *Proceedings of the 26th Conference in Uncertainty in Artificial Intelligence (UAI)*.

Hauskrecht, M. (2000). Value-function approximations for partially observable markov decision processes. *J. of Artificial Intelligence Research*, 13(1):33–94.

Hollinger, G. and Sukhatme, G. (2013). Stochastic motion planning for robotic information gathering. In *Robotics: Science and Systems (RSS)*.

Huang, G., Kaess, M., and Leonard, J. J. (2014). Consistent unscented incremental smoothing for multi-robot cooperative target tracking. *Robotics and Autonomous Systems*.

Huang, S., Kwok, N., Dissanayake, G., Ha, Q., and Fang, G. (2005). Multi-step look-ahead trajectory planning in SLAM: Possibility and necessity. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 1091–1096.

Indelman, V., Carlone, L., and Dellaert, F. (2013). Towards planning in generalized belief space. In *Proc. of the Intl. Symp. of Robotics Research (ISRR)*.

Indelman, V., Carlone, L., and Dellaert, F. (2014). Planning under uncertainty in the continuous domain: a generalized belief space approach. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*.

Kaelbling, L., Littman, M., and Cassandra, A. (1998). Planning and acting in partially observable stochastic domains. 101:99–134.

Kaelbling, L. and Lozano-Pérez, T. (2011). Pre-image backchaining in belief space for mobile manipulation. In *Proc. of the Intl. Symp. of Robotics Research (ISRR)*.

Kaelbling, L. and Lozano-Pérez, T. (2012). Unifying perception, estimation and action for mobile manipulation via belief space planning. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 2952–2959.

Kaelbling, L. and Lozano-Pérez, T. (2013). Integrated task and motion planning in belief space. *Intl. J. of Robotics Research*.

Kaess, M. and Dellaert, F. (2009). Covariance recovery from a square root information matrix for data association. *Robotics and Autonomous Systems*.

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J., and Dellaert, F. (2011). iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Shanghai, China.

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J., and Dellaert, F. (2012). iSAM2: Incremental smoothing and mapping using the Bayes tree. *Intl. J. of Robotics Research*, 31:217–236.

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

Kim, A. and Eustice, R. (2013). Perception-driven navigation: Active visual SLAM for robotic area coverage. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*.

Konolige, K., Grisetti, G., Kuemmerle, R., Burgard, W., Benson, L., and Vincent, R. (2010). Efficient sparse pose adjustment for 2D mapping. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 22–29, Taipei, Taiwan.

Kontitsis, M., Theodorou, E., and Todorov, E. (2013). Multi-robot active slam with relative entropy optimization. In *American Control Conference*.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). g2o: A general framework for graph optimization. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China.

Leung, C., Huang, S., and Dissanayake, G. (2006). Active SLAM using model predictive control and attractor based exploration. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 5026–5031.

Martinez-Cantin, 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.

Martinez-Cantin, R., Freitas, N. D., Doucet, A., and Castellanos, J. (2007). Active policy learning for robot planning and exploration under uncertainty. In *Robotics: Science and Systems (RSS)*.

Minka, T. (1998). Expectation-Maximization as lower bound maximization. Tutorial available at http://research.microsoft.com/en-us/um/people/minka/papers/em.html.

Patil, S., Kahn, G., Laskey, M., Schulman, J., Goldberg, K., and Abbeel, P. (2014). Scaling up gaussian belief space planning through covariance-free trajectory optimization and automatic differentiation. In *Intl. Workshop on the Algorithmic Foundations of Robotics*.

Platt, R., Tedrake, R., Kaelbling, L., and Lozano-Pérez, T. (2010). Belief space planning assuming maximum likelihood observations. In *Robotics: Science and Systems (RSS)*, pages 587–593.

Potthast, C. and Sukhatme, G. (2011). Next best view estimation with eye in hand camera. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*.

Prentice, S. and Roy, N. (2009). The belief roadmap: Efficient planning in belief space by factoring the covariance. *Intl. J. of Robotics Research*, 28(11-12):1448–1465.

Silver, D. and Veness, J. (2010). Monte-carlo planning in large pomdps. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2164–2172.

Sim, R. and Roy, N. (2005). Global A-optimal robot exploration in SLAM. pages 661–666.

Singh, A., Krause, A., Guestrin, C., and Kaiser, W. (2009). Efficient informative sensing using multiple robots. *J. of Artificial Intelligence Research*, 34:707–755.

Sommer, H., Pradalier, C., and Furgale, P. (2013). Automatic differentiation on differentiable manifolds as a tool for robotics. In *Proc. of the Intl. Symp. of Robotics Research (ISRR)*.

Stachniss, C., Grisetti, G., and Burgard, W. (2005). Information gain-based exploration using rao-blackwellized particle filters. In *Robotics: Science and Systems (RSS)*, pages 65–72.

Stachniss, C., Haehnel, D., and Burgard, W. (2004). Exploration with active loop-closing for FastSLAM. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*.

Strasdat, H., Montiel, J. M. M., and Davison, A. J. (2010). Real-time monocular SLAM: Why filter? In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 2657–2664.

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. The MIT press, Cambridge, MA.

Toussaint, M. (2009). Pros and cons of truncated gaussian ep in the context of approximate inference control. In *NIPS-Workshop on Probabilistic Approaches for Robotics and Control*, volume 21.

Valencia, R., Miró, J. V., Dissanayake, G., and Andrade-Cetto, J. (2011). Active pose SLAM. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 1885–1891.

Valencia, R., Morta, M., Andrade-Cetto, J., and Porta, J. (2013). Planning reliable paths with pose SLAM. *IEEE Trans. Robotics*.

Van Den Berg, J., Patil, S., and Alterovitz, R. (2012). Motion planning under uncertainty using iterative local optimization in belief space. *Intl. J. of Robotics Research*, 31(11):1263–1278.