Autonomous Robots
https://doi.org/10.1007/s10514-022-10045-w

Bayesian incremental inference update by re-using calculations from belief space planning: a new paradigm

Elad I. Farhi[1] · **Vadim Indelman**[2]

Received: 5 January 2021 / Accepted: 17 May 2022

Abstract
Inference and decision making under uncertainty are key processes in every autonomous system and numerous robotic problems. In recent years, the similarities between inference and decision making triggered much work, from developing unified computational frameworks to pondering about the duality between the two. In spite of these efforts, inference and control, as well as inference and belief space planning (BSP) are still treated as two separate processes. In this paper we propose a paradigm shift, a novel approach which deviates from conventional Bayesian inference and utilizes the similarities between inference and BSP. We make the *key observation* that inference can be efficiently updated using predictions made during the decision making stage, even in light of inconsistent data association between the two. We developed a two staged process that implements our novel approach and updates inference using calculations from the precursory planning phase. Using autonomous navigation in an unknown environment along with `iSAM2` efficient methodologies as a test case, we benchmarked our novel approach against standard Bayesian inference, both with synthetic and real-world data (KITTI dataset). Results indicate that not only our approach improves running time by at least a factor of two while providing the same estimation accuracy, but it also alleviates the computational burden of state dimensionality and loop closures.

Keywords Artificial intelligence · Simultaneous localization And mapping (SLAM) · Passive-SLAM · Active-SLAM · Bayesian incremental inference · Belief space planning (BSP) · Inference update · Inference and planning · Joint inference and planning

1 Introduction

Real life scenarios in autonomous systems and artificial intelligence involve agent(s) that are expected to reliably and efficiently operate under different sources of uncertainty, often with limited knowledge regarding the environment; e.g. autonomous navigation and simultaneous localization and mapping (SLAM), search and rescue scenarios, object manipulation and robot-assisted surgery. These settings necessitate probabilistic reasoning regarding high dimensional problem-specific states. For instance, in SLAM, the state typically represents robot poses and mapped static or dynamic landmarks, while in environmental monitoring and other sensor deployment related problems the state corresponds to an environmental field to be monitored (e.g. temperature as a function of position and perhaps time).

Attaining these levels of autonomy involves two key processes, inference and decision making under uncertainty. The former maintains a belief regarding the high-dimensional state given available information thus far, while the latter, also often referred to as belief space planning (BSP), is entrusted with determining the next best action(s).

The inference problem, has been addressed by the research community extensively over the past decades. In particular, focus was given to inference over high-dimensional state spaces with SLAM being a representative problem, and to computational efficiency to facilitate online operation, as required in numerous robotics systems. Over the years, the solution paradigm for the inference problem has

This research was partially supported by the Israel Science Foundation (ISF) and by the Israel Ministry of Science and Technology (MOST).

✉ Elad I. Farhi
eladfarhi@gmail.com

Vadim Indelman
vadim.indelman@technion.ac.il

[1] Technion Autonomous Systems Program (TASP), Technion - Israel Institute of Technology, 32000 Haifa, Israel

[2] Department of Aerospace Engineering, Technion - Israel Institute of Technology, 32000 Haifa, Israel

evolved. From EKF based methods Davison et al. (2007); Haykin (2001), through information form recursive Thrun et al. (2004) and smoothing methods Dellaert and Kaess (2006); Eustice et al. (2006), and in recent years up to incremental smoothing approaches, such as `iSAM` Kaess et al. (2008) and `iSAM2` Kaess et al. (2012).

Given the posterior belief from the inference stage, decision making under uncertainty and belief space planning approaches are entrusted with providing the next optimal action sequence given a certain objective. The aforementioned is accomplished by reasoning about belief evolution for different candidate actions while taking into account different sources of uncertainty. The corresponding problem is an instantiation of a partially observable Markov decision process (POMDP) problem, known as PSAPCE-complete Papadimitriou and Tsitsiklis (1987), hence computationally intractable for all but the smallest problems, i.e. no more than a few dozen states Kaelbling et al. (1998).

Over the years, numerous approaches have been developed to trade-off suboptimal performance with reduced computational complexity of POMDP, see e.g. Pineau et al. (2006), Kurniawati et al. (2008), Hollinger and Sukhatme (2014), Toussaint (2009). While the majority of these approaches, including Prentice and Roy (2009), Platt et al. (2010), Bry and Roy (2011), Van Den Berg et al. (2012), assumed some sources of absolute information (GPS, known landmarks) are available or considered the environment to be known, recent research relaxed these assumptions, accounting for the uncertainties in the mapped environment thus far as part of the decision making process Kim and Eustice (2014); Indelman et al. (2015) at the price of increased state dimensionality.

A crucial component in both inference and BSP is data association (DA), i.e. associating between sensor observations and the corresponding landmarks. Incorrect DA in inference or BSP can lead to catastrophic failures, due to wrong estimation in inference or incorrect belief propagation within BSP that would result in incorrect, and potentially unsafe, actions. Recent research thus focused on developing approaches that are robust to incorrect DA, considering both passive Carlone et al. (2014); Indelman et al. (2016); Olson and Agarwal (2013); Sunderhauf and Protzel (2012) and active perception Pathak et al. (2018).

Regardless of the decision making approach being used, in order to determine the next (sub)optimal actions the current belief is propagated using various action sequences. The propagated beliefs are then solved in order to provide an objective function value, thus enabling to determine the (sub)optimal actions. Solving a propagated belief is equivalent to performing inference over the belief, hence solving multiple inference problems is inevitable when trying to determine the (sub)optimal actions.

However, despite the similarities between inference and decision making, the two problems have been typically treated separately. Only in recent years, the research community has started investigating and exploiting these similarities between the two processes. For example, Kobilarov et al. (2015) and Ta et al. (2014) developed Differential Dynamic Programming (DDP) and Factor Graph (FG) based unified computational frameworks, respectively, for inference and decision making. Toussaint and Storkey (2006) provided an approximate solution to Markov Decision Process (MDP) problem using inference optimization methods, and Todorov (2008) investigated the duality between optimal control and inference for MDP case. Despite these research efforts, inference and BSP are still being handled as two separate processes.

Our *key observation* is that similarities between inference and decision making paradigms could be utilized in order to save valuable computation time. Our approach is rooted in the joint inference and belief space planning concept, presented in Farhi and Indelman (2017) and Farhi and Indelman (2019b), which strives to handle both inference and decision making in a partially observable setting within a unified framework, to enable sharing and updating similar calculations across inference and planning (see discussion in Sect. 5). In contrast to the notion of joint inference and control, which considers an MDP setting, we consider a partially observable setting (POMDP). Through the symbiotic relation enabled by considering the joint inference and BSP problems we make the following key research hypothesis: Inference can be efficiently updated using a precursory planning stage. This paper investigates this novel concept for inference update, considering operation in uncertain or unknown environments and compares it against the current state of the art in both simulated and real-life environments.

Updating inference with a precursory planning stage can be considered as a deviation from conventional Bayesian inference. Rather than updating the belief from the previous time instant with new incoming information (e.g. measurements), we propose to exploit the fact that similar calculations have already been performed within planning, in order to appropriately update the belief in inference more efficiently. We denote this novel approach by Re-Use BSP inference, or `RUB inference` in short.

The standard plan-act-infer framework of a typical autonomous system with conventional Bayesian approach for inference update is presented in Fig. 1a. First, BSP determines the next best action(s) given the posterior belief at current time; the robot performs this action(s); information is gathered and the former belief from the precursory inference is updated with new information (sensor measurements); the new posterior belief is then transferred back to the planning block in order to propagate it into future beliefs and provide again with the next action(s).

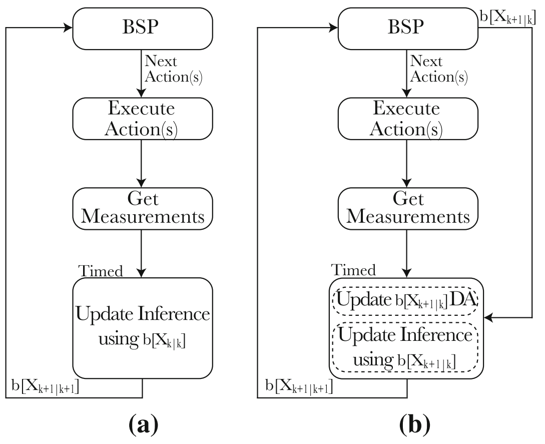


Fig. 1 High level algorithm for joint inference and BSP presented in a block diagram: **a** presents a standard plan-act-infer framework with Bayesian inference and BSP treated as separate processes; **b** presents our novel approach for inference update using precursory planning. Instead of updating the belief from precursory inference with new information we propose to update the belief from a precursory planning phase. Since the only difference between (a) and (b) manifests in computation time within the inference block, it is timed for comparison

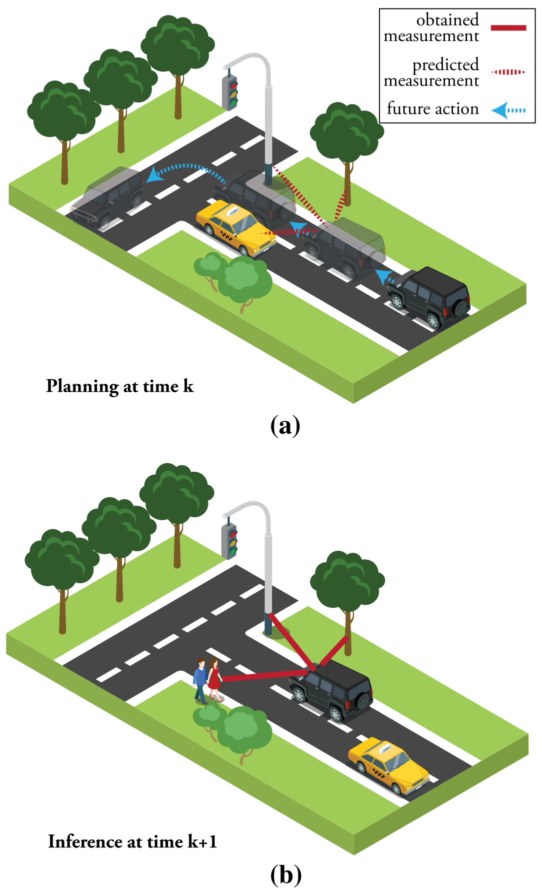


Fig. 2 Illustration for inconsistent DA between planning and succeeding inference: **a** at time k, our robot (i.e. the black jeep) plans three steps into the future. For the future step $k + 1$ it predicts measurements from three landmarks (tree, traffic-light and taxi). **b** After executing the first action our robot obtained three measurements from the environment. Two of them (i.e. tree and traffic-light) match the predicted DA from precursory planning session, while the third is associated to a new landmark (i.e. the couple that came out of the taxi)

Our proposed concept, `RUB inference`, is presented in Fig. 1b. `RUB inference` differs from the conventional Bayesian inference in two aspects: The output of the BSP process and the procedure of inference update. As opposed to standard Bayesian inference, in `RUB inference`, BSP output includes the next action(s) as well as the corresponding propagated future beliefs, no other changes are required in BSP in order to facilitate `RUB inference`. These beliefs are used to update inference while potentially taking care of data association aspects, rather than using the belief from precursory inference as conventionally done under Bayesian inference. As can be seen in Fig. 1b, the inference block contains data association (DA) update before the actual inference update. There are a lot of elements that can cause the DA in planning to be partially different than the DA established in the successive inference, e.g. estimation errors, disturbances, and dynamic or un-modeled unseen environments.

We start investigating this novel concept under a simplifying assumption that the DA considered in planning is consistent to that acquired during the succeeding inference, e.g. we predicted an association to a specific previously mapped landmark and later indeed observed that landmark. Since data association only relates to connections between variables and not to the measurement value, we are left with replacing the (potentially) incorrect measurement values, used within planning, with the actual values. Under this assumption, we provide four exact methods to efficiently update inference using the belief calculated by the precursory planning phase. As will be seen, these methods provide the same estimation accuracy as the conventional Bayesian inference approach, with a significantly shorter computation time.

We later relax the simplifying assumption mentioned above, and show inference can be efficiently updated using the precursory planning stage even when the DA considered in the two processes is partially different. Figure 2 illustrates such a case of inconsistent DA using a simple navigation problem. At time k our automated car (denoted by a black jeep), performs planning with a horizon of three steps. Figure 2a presents the chosen candidate action sequence along with the predicted measurements for future time $k + 1$. Our automated car predicts that at future time $k+1$ it would obtain measurements from the tree, the traffic-light and the taxi from the opposite lane. In addition to association, these predicted measurements also have values (e.g. pixels, distance) which depend on the state estimation (of both robot position and landmarks). Under an MPC framework, Fig. 2b presents the succeeding inference for current time $k+1$, in which our automated car advanced a bit more than planned, and indeed obtained three measurements. Two of these measurements are to the tree and the traffic-light (i.e. with consistent DA), while the third is to the couple that left the taxi (i.e. inconsistent DA). In such a case, merely updating the measurement values will not resolve the difference between the aforementioned DAs; instead the DA should be updated to match the acquired data, before updating the measurement values. We provide a novel paradigm to update inconsistent DA, leveraging `iSAM2` graphical model based methodologies, thus setting the conditions for complete inference update via BSP regardless of DA consistency.

To summarize, our contributions in this paper[1] are as follows: (*a*) We introduce `RUB inference`, a novel approach for saving computation time during the inference stage by reusing calculations made during the precursory planning stage; (*b*) We provide four exact methods, that utilize our concept under the assumption of consistent DA. We evaluate these four methods and compare them to the state of the art in simulation. (*c*) We provide a paradigm for incrementally updating inconsistent DA, thereby relaxing the afore-mentioned assumption; (*d*) We evaluate our complete paradigm and compare it to the state of the art both in simulation and on real-world data, considering the problem of autonomous navigation in unknown environments.

This paper is organized as follows. Section 2 formulates the discussed problem. Section 3 presents the suggested approach and its mathematical formulation. Section 4 presents a thorough analysis of the suggested approach and a comparison to related work. Section 5 discusses a broader perspective of `RUB Inference`. Section 6 captivates the conclusions of our work along with possible extensions and usage. To improve coherence, several aspects are covered in appendices.

2 Background and problem formulation

In this work, we consider the joint inference and belief space planning problem in a model predictive control (MPC) setting, i.e. BSP is performed after each inference phase. This problem can be roughly divided into two successive and recursive stages, namely inference and planning. The former performs inference given all information up to current time, updating the belief over the state with incoming information (e.g. sensor measurements). The latter produces the next control action(s), given the belief from the former inference stage and a user defined objective function.

[1] A preliminary version of this paper appeared in Farhi and Indelman (2017).

Let x_t denote the robot's state at time instant t and $\mathcal{L}$ represent the world state if the latter is uncertain or unknown. For example, for SLAM problem, it could represent objects or 3D landmarks. The joint state, up to time k, is defined as

$$X_k = \{x_0, \ldots, x_k, \mathcal{L}\} \in \mathbb{R}^n. \tag{1}$$

We shall be using the notation $t|k$ to refer to some time instant t while considering information up to time k; as will be shown in the sequel, this notation will allow to refer to *sequential* inference and planning phases in a unified manner.

Let $z_{t|k}$ and $u_{t|k}$ denote, respectively, the measurements and the applied control action at time t, while the current time is k. For example, $z_{k+1|k}$ represents measurements from a future time instant $k+1$ while $z_{k-1|k}$ represents measurements from a past time instant $k-1$, with the present time being k in both cases. Representing the measurements and controls up to time t, given current time k, as

$$z_{1:t|k} \doteq \{z_{1|k}, \ldots, z_{t|k}\}\ ,\ u_{0:t-1|k} \doteq \{u_{0|k}, \ldots, u_{t-1|k}\}, \tag{2}$$

the posterior probability density function (pdf) over the joint state, denoted as the *belief*, is given by

$$b[X_{t|k}] \doteq \mathbb{P}(X_t|z_{1:t|k}, u_{0:t-1|k}). \tag{3}$$

For $t=k$, Eq. (3) represents the posterior at current time k, while for $t>k$ it represents planning stage posterior for a specific sequence of future actions and observations. Using Bayes rule, Eq. (3) can be rewritten as

$$\mathbb{P}(X_t|z_{1:t|k}, u_{0:t-1|k}) \propto \mathbb{P}(x_0)\cdot\prod_{i=1}^{t}\left[\mathbb{P}(x_i|x_{i-1}, u_{i-1|k})\prod_{j\in\mathcal{M}_{i|k}}\mathbb{P}(z^j_{i|k}|x_i, l_j)\right], \tag{4}$$

where $\mathbb{P}(X_0)$ is the prior on the initial joint state, $\mathbb{P}(x_i|x_{i-1}, u_{i-1|k})$ and $\mathbb{P}(z^j_{i|k}|x_i, l_j)$ denote, respectively, the motion and measurement likelihood models. The set $\mathcal{M}_{i|k}$ contains all landmark indices observed at time i, i.e. it denotes data association (DA). The measurement of some landmark j at time i is denoted by $z^j_{i|k} \in z_{i|k}$. Under graphical representation of the belief, the conditional probabilities of the motion and observation models as well as the prior, can be denoted as factors (see Appendix-B). Eq. (4) can also be represented by a multiplication of these factors

$$\mathbb{P}(X_t|z_{1:t|k}, u_{0:t-1|k}) \propto \prod_{i=0}^{t}\{f_j\}_{i|k}\ , \tag{5}$$

where $\{f_j\}_{i|k}$ represents all factors added at time i while current time is k. The motion and measurement models are conventionally modeled with additive zero-mean Gaussian noise

$$x_{i+1} = f(x_i, u_i) + w_i \quad , \quad w_i \sim \mathcal{N}(0, \Sigma_w) \tag{6}$$
$$z_i^j = h(x_i, l_j) + v_i \quad , \quad v_i \sim \mathcal{N}(0, \Sigma_v), \tag{7}$$

where f and h are known possibly non-linear functions, Σ_w and Σ_v are the process and measurement noise covariance matrices respectively.

2.1 Inference

For the inference problem, $t \leq k$, i.e time instances that are equal or smaller than current time. The maximum a posteriori (MAP) estimate of the joint state X_k for time $t = k$ is given by

$$X^\star_{k|k} = \arg\max_{X_k} \; b[X_{k|k}] = \arg\max_{X_k} \mathbb{P}(X_k | z_{1:k|k}, u_{0:k-1|k}). \tag{8}$$

For the Gaussian case, the MAP solution produces the first two moments of the belief through solving a Non-linear Least Squares (NLS) problem, as will be shown later on. The MAP estimate from Eq. (8) is referred to as the *inference solution* in which, all controls and observations until time instant k are known.

2.2 Planning in the belief space

As mentioned, the purpose of planning is to determine the next optimal action(s). Finite horizon belief space planning for L look ahead steps involves inference over the beliefs

$$b[X_{k+l|k}] = \mathbb{P}(X_{k+l} | z_{1:k+l|k}, u_{0:k+l-1|k}), \quad l \in [k{+}1, k{+}L] \tag{9}$$

where we use the same notation as in Eq. (3) to denote the current time is k. The belief (9) can be written recursively as a function of the belief $b[X_{k|k}]$ from the inference phase as

$$b[X_{k+l|k}] = b[X_{k|k}] \cdot \prod_{i=k+1}^{k+l} \left[\mathbb{P}(x_i | x_{i-1}, u_{i-1|k}) \prod_{j \in \mathcal{M}_{i|k}} \mathbb{P}(z^j_{i|k} | x_i, l_j) \right], \tag{10}$$

for the considered action sequence $u_{k:k+l-1|k}$ at planning time k, and observations $z_{k+1:k+l|k}$ that are expected to be obtained upon execution of these actions. The set $\mathcal{M}_{i|k}$ denotes landmark indices that are expected to be observed at a future time instant i. It is worth stressing that the future belief (10) is determined by a specific realization of unknown future observations $z_{k+1:k+l|k}$, as stated in the belief definition in (9). Since terms for future belief of the form $\mathbb{P}(X_{k+l} | z_{1:k+l|k}, u_{0:k+l-1|k})$ will be used frequently in this paper in order not to burden the reader we use the more compact form $b[X_{i|k}]$. Whenever $i > k$ the reader should consider the belief $b[X_{i|k}]$ as a function of a specific realization of future observations.

One can now define a general objective function

$$J(u_{k:k+L-1|k}) \doteq \mathop{\mathbb{E}}_{z_{k+1:k+L|k}} \left[\sum_{i=k+1}^{k+L} c_i \left(b[X_{i|k}], u_{i-1|k} \right) \right], \tag{11}$$

with immediate costs (or rewards) c_i and where the expectation considers all the possible realizations of the future observations $z_{k+1:k+L|k}$. Conceptually, one could also reason whether these observations will actually be obtained, e.g. by considering also different realizations of $\mathcal{M}_{i|k}$. Note that for Gaussian distributions considered herein and information-theoretic costs (e.g. entropy), it can be shown that the expectation operator can be omitted under maximum-likelihood observations assumption Indelman et al. (2015), while another alternative is to simulate future observations via sampling, e.g. (Farhi and Indelman, 2019a, Sect. II-B), if such a simulator is available. The optimal open-loop control can now be defined as

$$u^\star_{k:k+L-1|k} = \arg\min_{u_{k:k+L-1|k}} J(u_{k:k+L-1|k}). \tag{12}$$

Evaluating the objective function (11) for a candidate action sequence involves calculating belief evolution for the latter, i.e. solving the inference problem for each candidate action using predicted future associations and measurements. Note that since we consider an MPC framework, the optimal control is affectively not an open-loop control, since it is being recalculated at each single action step.

2.3 Problem statement

Our key observation is that inference and BSP share similar calculations. Despite the similarities between them, they are treated as separate processes, thus duplicating costly calculations and increasing valuable computation time. This observation is impervious to any specific paradigms used for inference or planning and constitutes the difference between the use of `RUB inference` as opposed to conventional Bayesian inference.

Our goal is to salvage valuable computation time in the inference update stage by exploiting the similarities between inference and precursory planning, thus without affecting solution accuracy or introducing new assumptions.

3 Approach

Calculating the next optimal action $u^{\star}_{k|k} \in u^{\star}_{k:k+L-1|k}$ within BSP necessarily involves inference over the belief $b[X_{k+1|k}]$ conditioned on the same action $u^{\star}_{k|k}$. As we discuss in the sequel, this belief $b[X_{k+1|k}]$ can be different than $b[X_{k+1|k+1}]$ (the posterior at current time $k+1$) due to partially inconsistent data association and difference between measurement values considered in planning and those obtained in practice in inference. Our approach for `RUB Inference`, takes care of both of these aspects, thereby enabling to obtain $b[X_{k+1|k+1}]$ from $b[X_{k+1|k}]$.

In the following, we first analyze the similarities between inference and BSP (Sects. 3.1 and 3.2), and use these insights in Sect. 3.4 to develop methods for inference update under a simplifying assumption of consistent DA. We then relax this assumption, by analyzing the possible scenarios for inconsistent DA between inference and precursory planning (Sect. 3.5.1), and deriving a method for updating inconsistent DA (Sect. 3.5.2).

It is worth stressing that the only thing needed to be changed in any BSP algorithm in order to support our paradigm for `RUB Inference`, is just adding more information to its output. More specifically, outputting not only the (sub)optimal action $u^{\star}_{k|k}$, but also the corresponding future belief $b[X_{k+1|k}]$ (e.g. the difference between Figs. 1a,b).

3.1 Looking into inference

To better understand the similarities between inference and precursory planning, let us break down the inference solution to its components. Introducing Eqs. (4–7) into Eq. (8) and taking the negative logarithm yields the following non-linear least squares problem (NLS)

$$X^{\star}_{k|k} = \underset{X_k}{\arg\min} \|x_0 - x^{\star}_0\|^2_{\Sigma_0} + \sum_{i=1}^{k} \left[\|x_i - f(x_{i-1}, u_{i-1|k})\|^2_{\Sigma_w} + \sum_{j \in \mathcal{M}_{i|k}} \|z^j_{i|k} - h(x_i, l_j)\|^2_{\Sigma_v} \right], \tag{13}$$

where $\|a\|^2_{\Sigma} \doteq a^T \Sigma^{-1} a$ is the squared Mahalanobis norm.

Linearizing each of the terms in Eq. (13) and performing standard algebraic manipulations (see Appendix-A for derivation) yields

$$\Delta X^{\star}_{k|k} = \underset{\Delta X_k}{\arg\min} \|A_{k|k} \Delta X_k - b_{k|k}\|^2, \tag{14}$$

where $A_{k|k} \in \mathbb{R}^{m \times n}$ is the Jacobian matrix and $b_{k|k} \in \mathbb{R}^m$ is the right hand side (RHS) vector. In a more elaborated representation

$$A_{k|k} = \begin{bmatrix} \Sigma_0^{-\frac{1}{2}} \\ \mathcal{F}_{1:k|k} \\ \mathcal{H}_{1:k|k} \end{bmatrix} \quad , \quad b_{k|k} = \begin{bmatrix} 0 \\ \breve{b}^{\mathcal{F}}_{1:k|k} \\ \breve{b}^{\mathcal{H}}_{1:k|k} \end{bmatrix}, \tag{15}$$

where $\mathcal{F}_{1:k|k}$, $\mathcal{H}_{1:k|k}$, $\breve{b}^{\mathcal{F}}_{1:k|k}$ and $\breve{b}^{\mathcal{H}}_{1:k|k}$ (see Appendix-A) denote the Jacobian matrices and RHS vectors of all motion and observation terms accordingly, for time instances $1:k$ when the current time is k. These Jacobians, along with the corresponding RHS can be referred to by

$$\mathcal{A}_{1:k|k} = \begin{bmatrix} \mathcal{F}_{1:k|k} \\ \mathcal{H}_{1:k|k} \end{bmatrix} \quad , \quad \breve{b}_{1:k|k} = \begin{bmatrix} \breve{b}^{\mathcal{F}}_{1:k|k} \\ \breve{b}^{\mathcal{H}}_{1:k|k} \end{bmatrix}, \tag{16}$$

While there are a few methods to solve Eq. (14), we choose QR factorization as presented, e.g., in Kaess et al. (2008). The QR factorization of the Jacobian matrix $A_{k|k}$ is given by the orthonormal rotation matrix $Q_{k|k}$ and the upper triangular matrix $R_{k|k}$

$$A_{k|k} = Q_{k|k} R_{k|k}. \tag{17}$$

Eq. (17) is introduced into Eq. (14), thus producing

$$R_{k|k} \Delta X_k = d_{k|k}, \tag{18}$$

where $R_{k|k}$ is un upper triangular matrix and $d_{k|k}$ is the corresponding RHS vector, given by the original RHS vector and the orthonormal rotation matrix $Q_{k|k}$

$$d_{k|k} \doteq Q^T_{k|k} b_{k|k}. \tag{19}$$

We can now solve Eq. (18) for ΔX_k via back substitution, update the linearization point, and repeat the process until convergence. Eq. (18) can also be presented using a Bayes tree (BT) Kaess et al. (2010). A BT is a graphical representation of a factorized Jacobian matrix (the square root information matrix) R and the corresponding RHS vector d, in the form of a directed tree. More on the formulation of inference using graphical models can be found in Appendix-B. One can substantially reduce running time by exploiting sparsity and updating the QR factorization from the previous step with new information instead of calculating a factorization from scratch, see e.g. iSAM2 algorithm Kaess et al. (2012).

Given the inference solution, the belief $b[X_{k|k}]$ can be approximated by the Gaussian

$$b[X_{k|k}] \doteq \mathbb{P}(X_k | z_{1:k|k}, u_{0:k-1|k}) = \mathcal{N}(X^{\star}_{k|k}, \Lambda^{-1}_{k|k}), \tag{20}$$

while the information matrix is given by

$$\Lambda_{k|k} = A^T_{k|k} A_{k|k} = R^T_{k|k} R_{k|k}, \tag{21}$$

and the factorized Jacobian matrix $R_{k|k}$ along with the corresponding RHS vector $d_{k|k}$ can be used to update the linearization point and to recover the MAP estimate. In other words, the factorized Jacobian matrix $R_{k|k}$ and the corresponding RHS vector $d_{k|k}$ are sufficient for performing a single iteration within Gaussian belief inference.

3.2 Looking into planning

An interesting insight, that will be exploited in the sequel, is that the underlying equations of BSP are similar to those seen in Sect. 3.1. In particular, evaluating the belief at the Lth look ahead step, $b[X_{k+L|k}]$, involves MAP inference over a certain action sequence $u_{k:k+l-1|k}$ and future measurements $z_{k+1:k+l|k}$, which in turn, as in Sect. 3.1, can be described as an NLS problem

$$\begin{aligned} X^\star_{k+L|k} = \underset{X_{k+L}}{\arg\min} \|X_k - X^\star_{k|k}\|^2_{\Lambda^{-1}_{k|k}} &+ \sum_{i=k+1}^{k+L} \Bigg[\|x_i - f(x_{i-1}, u_{i-1|k})\|^2_{\Sigma_w} \\ &+ \sum_{j \in \mathcal{M}_{i|k}} \|z^j_{i|k} - h(x_i, l_j)\|^2_{\Sigma_v} \Bigg] \end{aligned} \tag{22}$$

For $i > k$, the set $\mathcal{M}_{i|k}$ contains *predicted* associations for future time instant i; hence, we can claim that $\forall i > k$ it is possible that $\mathcal{M}_{i|k} \neq \mathcal{M}_{i|i}$. In other words, it is possible that associations from the planning stage, $\mathcal{M}_{k+1|k}$, would be partially different than the associations from the corresponding inference stage $\mathcal{M}_{k+1|k+1}$. Moreover, the likelihood for inconsistent DA between planning and the corresponding inference rises as we look further into the future, i.e. with the distance $\|i-k\|$ increasing; e.g. $\mathcal{M}_{k+j|k}$ and $\mathcal{M}_{k+j|k+j}$ are less likely to be identical for $j = 10$ than they are for $j = 1$.

Predicting the unknown measurements $z_{k+1:k+L|k}$ in terms of both association and values can be done in various ways. In this paper the DA is predicted using current state estimation, and measurement values are obtained using the maximum-likelihood (ML) assumption, i.e. assuming zero innovation Dellaert and Kaess (2006). The robot pose is first propagated using the motion model (6). All landmark estimations are then transformed to the robot's new camera frame. Once in the robot camera frame, all landmarks that are within the robot's field of view are considered to be seen by the robot (predicted DA). The estimated position of each landmark, that is considered as visible by the robot, is being projected to the camera image plane Hartley and Zisserman (2004), thus generating measurements. It is worth mentioning that the aforementioned methodology is not able to predict occurrences of new landmarks, since it is based solely on the map the robot built thus far, i.e. current joint state estimation. The ability to predict occurrences of new landmarks would increase the advantage of `RUB Inference` over conventional Bayesian inference (as discussed in the sequel), hence is left for future work.

Once the predicted measurements are acquired, by following a similar procedure to the one presented in Sect. 3.1, for each action sequence we get

$$\Delta X^\star_{k+L|k} = \underset{\Delta X_{k+L}}{\arg\min} \|A_{k+L|k} \Delta X_{k+L} - b_{k+L|k}\|^2. \tag{23}$$

The Jacobian matrix $A_{k+L|k}$ and RHS vector $b_{k+L|k}$ are defined as

$$A_{k+L|k} \doteq \begin{bmatrix} A_{k|k} \\ \mathcal{A}_{k+1:k+L|k} \end{bmatrix}, \quad b_{k+L|k} \doteq \begin{bmatrix} b_{k|k} \\ \breve{b}_{k+1:k+L|k} \end{bmatrix}, \tag{24}$$

where $A_{k|k}$ and $b_{k|k}$ are taken from inference, see Eq. (14), and $\mathcal{A}_{k+1:k+L|k}$ and $\breve{b}_{k+1:k+L|k}$ correspond to the new terms obtained at the first L look ahead steps (e.g. see Eq. (16)). Note that although $\mathcal{A}_{k+1:k+L|k}$ is not a function of the (unknown) measurements $z_{k+1:k+L|k}$, it is a function of the predicted DA, $\mathcal{M}_{k+1:k+L|k}$ Indelman et al. (2015). Performing QR factorization, yields

$$A_{k+L|k} = Q^A_{k+L|k} R_{k+L|k}, \tag{25}$$

from which the information matrix, required in the information-theoretic cost, can be calculated. Using Eq. (24) the belief that correlates to the specific action sequence can be estimated, enabling evaluating the objective function (11). Determining the best action via Eq. (12) involves repeating this process for different candidate actions.

3.3 Similarities between inference and BSP

In an MPC setting, only the first action from the sequence $u^\star_{k:k+L-1|k}$ is executed, i.e.

$$u_{k|k+1} = u^\star_{k|k} \in u^\star_{k:k+L-1|k}. \tag{26}$$

In such case the difference between the belief obtained from BSP (for action $u^\star_{k|k}$)

$$b[X_{k+1|k}] \equiv \mathbb{P}(X_{k+1} | z_{1:k|k}, u_{0:k-1|k}, z_{k+1|k}, u^\star_{k|k}), \tag{27}$$

and the belief from the succeeding inference

$$b[X_{k+1|k+1}] \equiv \mathbb{P}(X_{k+1}|z_{1:k|k}, u_{0:k-1|k}, z_{k+1|k+1}, u_{k|k+1}), \tag{28}$$

is rooted in the set of measurements (i.e. $z_{k+1|k+1}$ vs. $z_{k+1|k}$), and the corresponding factors added at time instant $k+1$. These factor sets, denoted by $\{f_i\}_{k+1|k}$ and $\{f_j\}_{k+1|k+1}$ accordingly, can differ from one another in data association and measurement values. Since solving the belief requires linearization (14), it is important to note that both beliefs, $b[X_{k+1|k}]$ and $b[X_{k+1|k+1}]$, make use of the *same* initial linearization point $\bar{X}_{k+1}$ for the common variables. In particular, as in this work we do not reason within planning about new, unmapped thus far, landmarks, it follows that

$$X_{k+1|k} = \begin{bmatrix} X_{k|k} \\ x_{k+1} \end{bmatrix}, \quad X_{k+1|k+1} = \begin{bmatrix} X_{k|k} \\ x_{k+1} \\ L_{k+1}^{new} \end{bmatrix} \tag{29}$$

where L_{k+1}^{new} represents the new landmarks that were added to the belief for the first time at time instant $k+1$. The linearization point for the common variables is $[X_{k|k}^{\star}\ ,\ f(x_k, u_{k|k}^{\star})]$ for planning, and $[X_{k|k}^{\star}\ ,\ f(x_k, u_{k|k+1})]$ for succeeding inference, where $f(.)$ is the motion model (6). Since the (sub)optimal action provided by BSP is the one executed in the succeeding inference i.e. Eq. (26), the motion models are identical hence the same linearization point is used in both inference and precursory planning.

When considering the belief from planning (27), which is propagated with the next action (26) and predicted measurements, with the previously factorized form of $A_{k|k}$ and $b_{k|k}$, we get

$$A_{k+1|k}^{R} \doteq \begin{bmatrix} R_{k|k} \\ \mathcal{A}_{k+1|k} \end{bmatrix}, \quad b_{k+1|k}^{d} \doteq \begin{bmatrix} d_{k|k} \\ \breve{b}_{k+1|k} \end{bmatrix}. \tag{30}$$

Similarly, when considering the a posteriori belief from inference (28), propagated with the next action (26) and acquired measurements, with the previously factorized form of $A_{k|k}$ and $b_{k|k}$, we get

$$A_{k+1|k1}^{R} \doteq \begin{bmatrix} R_{k|k} \\ \mathcal{A}_{k+1|k+1} \end{bmatrix}, \quad b_{k+1|k+1}^{d} \doteq \begin{bmatrix} d_{k|k} \\ \breve{b}_{k+1|k+1} \end{bmatrix}. \tag{31}$$

For the same action (26), the difference between Eq. (30) to the equivalent representation of standard Bayesian inference (31) originates from the factors added at time $k+1$

$$\mathcal{A}_{k+1|k} \overset{?}{=} \mathcal{A}_{k+1|k+1}\ , \tag{32}$$

$$\breve{b}_{k+1|k} \overset{?}{=} \breve{b}_{k+1|k+1}\ . \tag{33}$$

Since the aforementioned share the same action sequence, the same linearization point and the same models, the differences remain limited to the DA and measurement values at time $k+1$.

In planning, DA is based on predicting which landmarks would be observed. This DA could very possibly be different than the actual landmarks the robot observes, as presented in Sect. 3.2. This inconsistency in DA manifests in both the Jacobian matrices and the RHS vectors. Even in case of consistent DA, the predicted measurements (if exist) would still be different than the actual measurements due to various reasons, e.g. the predicted position is different than the ground truth of the robot, measurement noise, inaccurate models.

While for consistent DA and the same linearization point Eq. (32) will always be true, the RHS vectors, specifically Eq. (33), would still be different due to the difference in measurement values considered in planning and actually obtained in inference.

It is worth stressing that consistent data association between inference and precursory planning suggests that all predictions for state variable (new or existing) associations were in fact true. In addition to the new robot state added each time instant, new variables could also manifest in the form of landmarks. Consistent DA implies that the future appearance of all new landmarks has been perfectly predicted during planning. Since for the purpose of this work, we use a simple prediction mechanism unable to predict new landmarks (see Sect. 3.2), consistent DA would inevitably mean no new landmarks in inference, i.e L_{k+1}^{new} is an empty set.

We start developing our method by assuming consistent DA between inference and precursory planning. In such a case the difference is limited to the RHS vectors. Later we relax this assumption by dealing with possible DA inconsistency prior to the update of the RHS vector, thus addressing the general and complete problem of inference update using `RUB Inference` paradigm.

3.4 Inference update from BSP assuming consistent data association

Let us assume that the DA between inference and precursory planning is consistent, whether the cause is a "lucky guess" during planning or whether the DA inconsistency has been resolved beforehand. Recalling the definition of $\mathcal{M}_{i|k}$ (see e.g. Eq. (10)), this assumption is equivalent to writing

$$\mathcal{M}_{k+1|k} \equiv \mathcal{M}_{k+1|k+1}. \tag{34}$$

In other words, landmarks considered to be observed at a future time $k+1$, will indeed be observed at that time. Note this does *not* necessarily imply that actual measurements and robot poses will be as considered within the planning stage, but it does necessarily state that both are considering the same variables and the same associations.

We now observe that the motion models in both $b[X_{k+1|k+1}]$ and $b[X_{k+1|k}]$ are evaluated considering the *same* control (i.e. the optimal control $u_k^{\star}$). Moreover, the robot pose x_{k+1}

is initialized to the *same* value in both cases as $f(x_k, u_k^\star)$, see e.g. Eq.(27) in Indelman et al. (2015), and thus the linearization point of all probabilistic terms in inference and planning is *identical*. This, together with the aforementioned assumption (i.e. Eq. (34) holds) allows us to write $A_{k+1|k} = A_{k+1|k+1}$, and hence

$$R_{k+1|k+1} \equiv R_{k+1|k}, \tag{35}$$

for the *first iteration* in the inference stage at time $k+1$.

Hence, in order to solve $b[X_{k+1|k+1}]$ we are left to find the RHS vector $d_{k+1|k+1}$, while $R_{k+1|k+1}$ can be *entirely re-used*.

In the sequel we present four methods that can be used for updating the RHS vector, and examine computational aspects of each. The four methods use two different approaches to update the RHS vector: while the first two (OTM and OTM-OO), utilize the rotation matrix available from factorization, the last two (DU and DU-OO) utilize information downdate / update principles. After we review the methods we shortly discuss the advantages and disadvantages of each (Sect. 3.4.5). It is worth stressing that each of these methods results in the same RHS vector which is also identical to the RHS vector that would have been obtained by the standard inference update. With both the factorized Jacobian matrix (i.e. R) and the RHS vector identical to the standard inference update approach, RUB inference provides the same estimation accuracy for the inference solution.

3.4.1 The orthogonal transformation matrix method - OTM

In the OTM method, we obtain $d_{k+1|k+1}$ following the definition as written in Eq. (19). Recall that at time $k+1$ in the inference stage, the posterior should be updated with new terms that correspond, for example, to motion model and obtained measurements. The RHS vector's augmentation, that corresponds to these new terms is denoted by $\breve{b}_{k+1|k+1}$, see Eq. (16). Given $R_{k|k}$ and $d_{k|k}$ from the inference stage at time k, the augmented system at time $k+1$ is

$$A^R_{k+1|k+1}\Delta X_{k+1} \doteq \begin{bmatrix} R_{k|k} \\ \mathcal{A}_{k+1|k+1} \end{bmatrix} \Delta X_{k+1} = \begin{bmatrix} d_{k|k} \\ \breve{b}_{k+1|k+1} \end{bmatrix} \tag{36}$$

which after factorization of $A^R_{k+1|k+1}$ (see Eqs. (17)-(19)) becomes

$$R_{k+1|k+1}\Delta X_{k+1} = d_{k+1|k+1}, \tag{37}$$

where

$$d_{k+1|k+1} = Q^T_{k+1|k+1} \begin{bmatrix} d_{k|k} \\ \breve{b}_{k+1|k+1} \end{bmatrix}. \tag{38}$$

As deduced from Eq. (38), the calculation of $d_{k+1|k+1}$ requires $Q_{k+1|k+1}$. Since $A^R_{k+1|k} \equiv A^R_{k+1|k+1}$ (see Sect. 3.4), we get $Q_{k+1|k+1} = Q_{k+1|k}$. However, $Q_{k+1|k}$, is already available from the precursory planning stage, see Eq. (25), and thus calculating $d_{k+1|k+1}$ via Eq. (38) does *not* involve QR factorization in practice. To summarize, under the OTM method we obtain the RHS vector $d_{k+1|k+1}$ in the following manner:

$$d_{k+1|k+1} = Q^T_{k+1|k} \begin{bmatrix} d_{k|k} \\ \breve{b}_{k+1|k+1} \end{bmatrix}. \tag{39}$$

where $Q^T_{k+1|k}$ is available from the factorization of precursory planning, $d_{k|k}$ is the RHS from inference at time k, and $\breve{b}_{k+1|k+1}$ are the new un-factorized RHS values obtained at time $k+1$ (Table 1).

3.4.2 The OTM - only observations method - OTM-OO

The OTM-OO method is a variant of the OTM method. OTM-OO aspires to utilize even more information from the planning stage. Since the motion models from inference and the precursory planning first step are identical, i.e. same function $f(.,.)$, see Eqs. (13) and (22), and as in both cases the *same* control is considered - Eq. (26), there is no reason to change the motion model data from the RHS vector $d_{k+1|k}$. In order to enable the aforementioned, we require the matching rotation matrix. One way would be to break down the planning stage as described in Sect. 3.2 into two stages, in which the motion and observation models are updated separately. Usually this breakdown is performed either way since a propagated future pose is required for predicting future measurements.

So following Sect. 3.4.1, instead of using $d_{k|k}$, we attain from planning the RHS vector already with the motion model ($d^{\mathcal{F}}_{k+1|k}$), augment it with the new measurements and rotate it with the corresponding rotation matrix obtained from the planning stage

$$d_{k+1|k+1} = Q^{\mathcal{H}^T}_{k+1|k} \begin{bmatrix} d^{\mathcal{F}}_{k+1|k} \\ \breve{b}^{\mathcal{H}}_{k+1|k+1} \end{bmatrix}. \tag{40}$$

The rotation matrix $Q^{\mathcal{H}}_{k+1|k}$ is given from the precursory planning stage where

$$Q^{\mathcal{H}}_{k+1|k} R_{k+1|k} = \begin{bmatrix} R^{\mathcal{F}}_{k+1|k} \\ \mathcal{H}_{k+1|k} \end{bmatrix}, \tag{41}$$

and where $R^{\mathcal{F}}_{k+1|k}$ is the factorized Jacobian propagated with the motion model given by

$$Q^{\mathcal{F}}_{k+1|k} R^{\mathcal{F}}_{k+1|k} = \begin{bmatrix} R_{k|k} \\ \mathcal{F}_{k+1|k} \end{bmatrix}. \tag{42}$$

Table 1 Notations for Sect. 3.4

Variable	Description
$\square_{t\|k}$	Of time t while current time is k
ΔX_k	State perturbation around linearization point
$\mathcal{M}_{t\|k}$	Data Association at time t
$A_{t\|k}$	Jacobian matrix at time t
$b_{t\|k}$	RHS vector at time t
$\mathcal{A}_{t\|k}$	Jacobian part related to all factors added at time t
$\mathcal{F}_{t\|k}$	Jacobian part related to motion factor added at time t
$\mathcal{H}_{t\|k}$	Jacobian part related to all factors added at time t without the motion factor
$\breve{b}_{t\|k}$	RHS vector related to all factors added at time t
$\breve{b}^{\mathcal{F}}_{t\|k}$	RHS vector related to motion factor at time t
$\breve{b}^{\mathcal{H}}_{t\|k}$	RHS vector related to all factors added at time t without the motion factor
$R_{t\|k}$	Factorized Jacobian, i.e. square root information matrix
$d_{t\|k}$	Factorized RHS vector
$A^R_{t\|k}$	Factorized $\left[R^T_{t-1\|k}, \mathcal{A}^T_{t\|k}\right]^T$
$R^{\mathcal{F}}_{t\|k}$	Factorized $\left[R^T_{t-1\|k}, \mathcal{F}^T_{t\|k}\right]^T$
$d^{\mathcal{F}}_{t\|k}$	Factorized $\left[d^T_{t-1\|k}, \breve{b}^{\mathcal{F}T}_{t\|k}\right]^T$
$R^{aug}_{t\|k}$	Factorized Jacobian at time $t-1$ zero padded to match factorized Jacobian at time t
$d^{aug}_{t\|k}$	Factorized RHS vector at time $t-1$ zero padded to natch factorize RHS vector at time t
$Q^A_{t\|k}$	Rotation matrix for factorizing $A_{t\|k}$ into $R_{t\|k}$
$Q_{t\|k}$	Rotation matrix for factorizing $A^R_{t\|k}$ into $R_{t\|k}$
$Q^{\mathcal{F}}_{t\|k}$	Rotation matrix for factorizing $\left[R^T_{t-1\|k}, \mathcal{F}^T_{t\|k}\right]^T$ into $R^{\mathcal{F}}_{t\|k}$
$Q^{\mathcal{H}}_{t\|k}$	Rotation matrix for factorizing $\left[R^{\mathcal{F}T}_{t\|k}, \mathcal{H}^T_{t\|k}\right]^T$ into $R_{t\|k}$

As will be seen later on, the OTM-OO method would prove to be the most computationally efficient between the four suggested methods.

3.4.3 The downdate update method - DU

In the DU method we propose to re-use the $d_{k+1|k}$ vector from the planning stage to calculate $d_{k+1|k+1}$.

While not necessarily required within the planning stage, $d_{k+1|k}$ could be calculated at that stage from $b_{k+1|k}$ and $Q_{k+1|k}$, see Eqs. (24)–(25). However, $b_{k+1|k}$ (unlike $A_{k+1|k}$) is a function of the unknown future observations $z_{k+1|k}$, which would seem to complicate things. Our solution to this issue is as follows: We assume *some* value for the observations $z_{k+1|k}$ and then calculate $d_{k+1|k}$ within the planning stage. As in inference at time $k+1$, the actual measurements $z_{k+1|k+1}$ will be different, we remove the contribution of $z_{k+1|k}$ to $d_{k+1|k}$ via information downdating (Sect. V-A Cunningham et al. 2013), and then appropriately incorporate $z_{k+1|k+1}$ to get $d_{k+1|k+1}$ using the same mechanism.

More specifically, downdating the measurements $z_{k+1|k}$ from $d_{k+1|k}$ is done via (Sect. V-A Cunningham et al. 2013)

$$d^{aug}_{k+1|k} = R^{aug^{-T}}_{k+1|k}\left(R^T_{k+1|k}d_{k+1|k} - \mathcal{A}^T_{k+1|k}\breve{b}_{k+1|k}\right), \tag{43}$$

where $\breve{b}_{k+1|k}$ is a function of $z_{k+1|k}$, see Eqs. (22)–(24), and where $R^{aug}_{k+1|k}$ is the downdated $R_{k+1|k}$ matrix which is given by

$$R^{aug^T}_{k+1|k}R^{aug}_{k+1|k} = A^{R^T}_{k+1|k}A^R_{k+1|k} - \mathcal{A}^T_{k+1|k}\mathcal{A}_{k+1|k}. \tag{44}$$

Interestingly, the above calculations are not really required: Since we already have $d_{k|k}$ from the previous inference stage, we can attain the downdated $d^{aug}_{k+1|k}$ vector more efficiently by augmenting $d_{k|k}$ with zero padding.

$$d^{aug}_{k+1|k} = \begin{bmatrix} d_{k|k} \\ 0 \end{bmatrix} \tag{45}$$

where $d^{aug}_{k+1|k}$ is the downdated RHS vector and 0 is a zero padding to match dimensions. Similarly, $R^{aug}_{k+1|k}$ can be calculated as

$$R^{aug}_{k+1|k} = \begin{bmatrix} R_{k|k} & 0 \\ 0 & 0 \end{bmatrix}, \tag{46}$$

where $R_{k|k}$ is zero padded to match dimensions of $R_{k+1|k}$.

Now, all which is left to get $d_{k+1|k+1}$, is to incorporate the new measurements $z_{k+1|k+1}$ (encoded in $\breve{b}_{k+1|k+1}$). We utilize the information downdating mechanism in (Cunningham et al., 2013, Sect. V-A), in order to update information. Intuitively, instead of downdating information from $d_{k+1|k}$, we would like to add information to $d^{aug}_{k+1|k}$. So by appropriately adjusting Eq.(43) this can be done via

$$d_{k+1|k+1} = R^{-T}_{k+1|k+1}(R^{aug^T}_{k+1|k} d^{aug}_{k+1|k} + \mathcal{A}^T_{k+1|k+1}\breve{b}_{k+1|k+1}), \tag{47}$$

where according to Eq. (34) $R_{k+1|k+1} \equiv R_{k+1|k}$ and $\mathcal{A}_{k+1|k+1} \equiv \mathcal{A}_{k+1|k}$, $R^{aug}_{k+1|k}$ is given by Eq.(46), $d^{aug}_{k+1|k}$ is given by Eq.(45), and $\breve{b}_{k+1|k+1}$ are the new un-factorized RHS values obtained at time $k+1$.

To summarize, under the DU method we obtain the RHS vector $d_{k+1|k+1}$ in the following manner:

$$d_{k+1|k+1} = R^{-T}_{k+1|k}\left(\begin{bmatrix} R_{k|k} & 0 \\ 0 & 0 \end{bmatrix}^T \begin{bmatrix} d_{k|k} \\ 0 \end{bmatrix} + \mathcal{A}^T_{k+1|k}\breve{b}_{k+1|k+1}\right). \tag{48}$$

3.4.4 The DU - only observations method - DU−OO

The DU−OO method is a variant of the DU method, where, similarly to Sect. 3.4.2, we utilize the fact that there is no reason to change the motion model data from the RHS vector $d_{k+1|k}$. Hence we would downdate all data with the exception of the motion model, and then update accordingly. As opposed to Sect. 3.4.3, now we do need to downdate using (Cunningham et al., 2013, Sect. V-A)

$$d^{\mathcal{F}}_{k+1|k} = R^{\mathcal{F}-T}_{k+1|k}(R^T_{k+1|k}d_{k+1|k} - \mathcal{H}^T_{k+1|k}\breve{b}^{\mathcal{H}}_{k+1|k}), \tag{49}$$

where $d^{\mathcal{F}}_{k+1|k}$ is the RHS vector, downdated from all predicted measurements with the exception of the motion model, and $R^{\mathcal{F}}_{k+1|k}$ is the equivalent downdated $R_{k+1|k}$ matrix which is given by

$$R^{\mathcal{F}^T}_{k+1|k}R^{\mathcal{F}}_{k+1|k} = A^{R^T}_{k+1|k}A^R_{k+1|k} - \mathcal{H}^T_{k+1|k}\mathcal{H}_{k+1|k}, \tag{50}$$

where $\mathcal{H}_{k+1|k}$ denotes the portion of the planning stage Jacobian, of the predicted factors with the exception of the motion model. Now, all which is left, is to update $d^{\mathcal{F}}_{k+1|k}$ with the new measurements from the inference stage

$$d_{k+1|k+1} = R^{-T}_{k+1|k+1}(R^{\mathcal{F}^T}_{k+1|k}d^{\mathcal{F}}_{k+1|k} + \mathcal{H}^T_{k+1|k+1}\breve{b}^{\mathcal{H}}_{k+1|k+1}), \tag{51}$$

where according to Eq. (34) $R_{k+1|k+1} \equiv R_{k+1|k}$ and $\mathcal{H}_{k+1|k+1} \equiv \mathcal{H}_{k+1|k}$, $R^{\mathcal{F}}_{k+1|k}$ is given by Eq.(50), $d^{\mathcal{F}}_{k+1|k}$ is given by Eq.(49), and $\breve{b}_{k+1|k+1}$ are the new un-factorized RHS values obtained at time $k+1$.

By introducing Eqs. (49) into (51) we can also avert from calculating $R^{\mathcal{F}}_{k+1|k}$ so under the DU−OO assumption we obtain the RHS vector $d_{k+1|k+1}$ in the following manner:

$$\begin{aligned} d_{k+1|k+1} \\ = R^{-T}_{k+1|k}\left(R^T_{k+1|k}d_{k+1|k} + \mathcal{H}^T_{k+1|k}\left(\breve{b}^{\mathcal{H}}_{k+1|k+1} - \breve{b}^{\mathcal{H}}_{k+1|k}\right)\right), \end{aligned} \tag{52}$$

which can be rewritten as

$$d_{k+1|k+1} = d_{k+1|k} + R^{-T}_{k+1|k}\mathcal{H}^T_{k+1|k}\left(\breve{b}^{\mathcal{H}}_{k+1|k+1} - \breve{b}^{\mathcal{H}}_{k+1|k}\right). \tag{53}$$

3.4.5 Discussion - RHS update methods

In this section we would like to give the reader some intuition regarding the advantages and disadvantages of the OTM approach when compared to the DU approach. Since both provide the same desired solution, the difference between them would manifest in computation time and ease of use. In the sequel we cover both starting with the complexity of each.

Let us compare the complexity required for updating the RHS by OTM, see Eq. (39), against the complexity required for updating the RHS by DU, see Eq. (48). For OTM we have a single multiplication between a sparse rotation matrix $Q_{k+1|k}$ and a vector, both in the dimension of the joint state at time k plus the number of rows of the linearized new factors (i.e. depending on number of factors and their types). The complexity of OTM would be given by the number of non zeros in the rotation matrix $Q_{k+1|k}$. In Appendix-C we provide some understanding on the creation of the rotation matrix $Q_{k+1|k}$, and also develop an expression for the number of non zeros in $Q_{k+1|k}$. We direct the reader to Fig. 3 for illustration of the new notations used in this discussion. Following the development in Appendix-C, the number of non zeros in $Q_{k+1|k}$ is represented by two potentially dominant terms separated by a simple condition

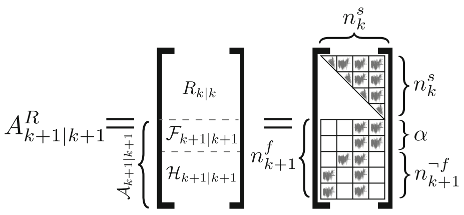


Fig. 3 Illustration of the Jacobian matrix $A^R_{k+1|k+1}$ introduced in Eq. (31), on its components and dimensions. These notations are used along Sect. 3.4.5, and brought here for the reader's convenience

$$O(\texttt{OTM}) = \begin{cases} O\left(\left(n^s_k + n^f_{k+1} - j\right)^2\right) & n^f_{k+1} \geq n^s_k \geq 6 \\ O\left(n^s_k \cdot n^f_{k+1}\right) & n^f_{k+1} < n^s_k \end{cases} \tag{54}$$

where j denotes the column index of the left-most entry in $\mathcal{A}_{k+1|k+1}$, n^s_k denotes the size of the joint state vector at the precursory time k, n^f_{k+1} denotes the number of rows in the linearized new factors $\mathcal{A}_{k+1|k+1}$. The condition in Eq. (54) is a simple upper bound to the real expression (see Eq. (93)), resulting with a cleaner condition without affecting the solution.

It is worth stressing that depending on its type, each state occupies more than a single row / column in the Jacobian, e.g. 6DOF robot pose occupies six rows and six columns. Similarly, depending on its type, each factor occupies more than a single row in the Jacobian, e.g. a monocular factor occupies two rows in the Jacobian.

For DU in addition to multiplications between upper triangular matrices and vectors, we have a matrix inverse. Differently from OTM here the matrix dimensions are of the joint state vector at time $k+1$, hence the worst case scenario for DU is a fully dense upper triangular matrix inverse

$$O(\texttt{DU}) = O\left({n^s_{k+1}}^2\right), \tag{55}$$

where n^s_{k+1} represents the size of the joint state vector at time $k+1$.

For the case of $n^f_{k+1} < n^s_k$ we should compare

$$n^s_k \cdot n^f_{k+1} \overset{?}{\lesseqgtr} n^s_{k+1} \cdot n^s_{k+1}. \tag{56}$$

Assuming states are not removed from the state vector, we can say

$$n^s_k \leq n^s_{k+1}, \tag{57}$$

then evidently

$$n^s_k \cdot n^f_{k+1} < n^s_{k+1} \cdot n^s_{k+1}. \tag{58}$$

For the case of $n^f_{k+1} \geq n^s_k \geq 6$ we should compare

$$n^s_k + n^f_{k+1} - j \overset{?}{\lesseqgtr} n^s_{k+1} \quad , \quad j \in [1, n^s_k], \tag{59}$$

so for this case OTM is computationally superior to DU if

$$n^f_{k+1} < n^s_{k+1} - n^s_k + j \quad , \quad j \in [1, n^s_k]. \tag{60}$$

It is worth stressing that unlike Eqs. (58), (60) is dependent on state ordering in the form of the left-most non zero entry in $\mathcal{A}_{k+1|k+1}$.

Concluding the complexity analysis of OTM and DU, OTM will be computationally superior to DU if the following holds

$$\left(n^f_{k+1} < n^s_k\right) \cup \left(n^f_{k+1} < n^s_{k+1} - n^s_k + j \quad \cap \quad n^f_{k+1} \geq n^s_k\right). \tag{61}$$

In other words, if the number of rows in $\mathcal{A}_{k+1|k+1}$ is smaller than the size of the state vector at time k OTM is computationally superior to DU. If the number of rows in $\mathcal{A}_{k+1|k+1}$ is larger or equal to the size of the state vector at time k, than OTM is computationally superior to DU only if the number of rows in $\mathcal{A}_{k+1|k+1}$ is smaller than the size of the added states at time $k+1$ plus the column index of the left-most state in $\mathcal{A}_{k+1|k+1}$.

Although most of the time DU is computationally inferior, unlike OTM that requires access to the rotation matrix which might not be easily available in every planning paradigm, DU makes use in a more readily available information: the inference solution of precursory time, the predicted factors, the new RHS vector at time $k+1$, and the factorized Jacobian from precursory planning. Therefore the advantage in using DU lies in the information availability with minimal adjustments to the planning stage.

Since OTM-OO would prove to perform the best empirically, let us get some intuition on why it is more efficient than OTM. The OO addition to OTM, refers to the use of the motion propagated belief $R^{\mathcal{F}}_{k+1|k}\ d^{\mathcal{F}}_{k+1|k}$ rather than the use of precursory inference solution $R_{k|k}\ d_{k|k}$. The dimension of $R^{\mathcal{F}}_{k+1|k}$ is larger from that of $R_{k|k}$ by a single robot pose, while the number of rows of $\mathcal{H}_{k+1|k+1}$ is smaller by a single robot pose from that of $\mathcal{A}_{k+1|k+1}$. Let us assume without affecting generality that our robot pose dimension is α. Under this assumption we can calculate Eq. (54) for both OTM and OTM-OO. Let $n^{\neg f}_{k+1}$ denote the number of rows of the newly added factors at time $k+1$ without the motion factor, i.e. $\mathcal{H}_{k+1|k+1}$ number of rows, so the complexity of OTM would be

$$O(\texttt{OTM}) = \begin{cases} O\left(\left(n^s_k + \left(n^{\neg f}_{k+1} + \alpha\right) - j\right)^2\right) & \left(n^{\neg f}_{k+1} + \alpha\right) \geq n^s_k \\ O\left(n^s_k \cdot \left(n^{\neg f}_{k+1} + \alpha\right)\right) = O\left(n^s_k \cdot n^{\neg f}_{k+1} + \alpha \cdot n^s_k\right) & \left(n^{\neg f}_{k+1} + \alpha\right) < n^s_k \end{cases}, \tag{62}$$

while the complexity of OTM-OO would be

$$O(\texttt{OTM-OO}) = \begin{cases} O\left(\left(\left(n_k^s + \alpha\right) + n_{k+1}^{\neg f} - j'\right)^2\right) & n_{k+1}^{\neg f} \geq \left(n_k^s + \alpha\right) \\ O\left(\left(n_k^s + \alpha\right) \cdot n_{k+1}^{\neg f}\right) = O\left(n_k^s \cdot n_{k+1}^{\neg f} + \alpha \cdot n_{k+1}^{\neg f}\right) & n_{k+1}^{\neg f} < \left(n_k^s + \alpha\right) \end{cases}, \tag{63}$$

where $j' \in [1, \left(n_k^s + \alpha\right)]$, opposed to $j \in [1, n_k^s]$. From comparing Eqs. (62 – 63), for the case where the size of added factors is larger than the state, we can deduce that other than the difference between j and j', they are the same. Judging the second case, we can see they differ by the difference between the size of the state at time k and the number of $\mathcal{A}_{k+1|k+1}$ rows. As we will see later on, OTM-OO empirically proves to be more efficient than OTM, which means that the state at time k is in fact larger than the number of size of $\mathcal{A}_{k+1|k+1}$ rows.

Revisiting Eq. (61) in-light of the understanding that the state at time k is in fact larger than the number of $\mathcal{A}_{k+1|k+1}$ rows we can say that OTM is computationally superior to DU without any restricting conditions.

3.5 Inconsistent data association

In order to address the more general and realistic scenario, the DA might require correction before proceeding to update the new acquired measurements. In the sequel we cover the possible scenarios of inconsistent data association and its graphical materialization, followed by a paradigm to update inconsistent DA from planning stage according to the actual DA attained in the consecutive inference stage. We later examine both the computational aspects and the sensitivity of the paradigm to various parameters both on simulated and real-life data.

3.5.1 Types of inconsistent DA

We would now discuss, without losing generality, the actual difference between the two aforementioned beliefs $b[X_{k+1|k}]$ and $b[X_{k+1|k+1}]$. As already presented in Sect. 3.4, in case of a consistent DA i.e. $\mathcal{M}_{k+1|k} = \mathcal{M}_{k+1|k+1}$, the difference between the two beliefs is narrowed down to the RHS vectors $d_{k+1|k}$ and $d_{k+1|k+1}$ which encapsulates the measurements $z_{k+1|k}$ and $z_{k+1|k+1}$ respectively. However, in the real world it is possible that the DA predicted in precursory planning would prove to be inconsistent to the DA attained in inference.

There are six possible scenarios representing the relations between DA in inference and precursory planning:

- In planning, association is assumed to either a new or existing variable, while in inference no measurement is received.
- In planning it is assumed there will be no measurement to associate to, while in inference a measurement is received and associated to either a new or existing variable.
- In planning, association is assumed to an existing variable, while in inference it is to a new variable.
- In planning, association is assumed to a new variable, while in inference it is to an existing variable.
- In planning, association is assumed to an existing variable, while in inference it is also to an existing variable (whether the same or not).
- In planning, association is assumed to a new variable, while in inference it is also to a new variable (whether the same or not) (Table 2).

While the first four bullets always describe inconsistent DA situations (e.g. in planning we assumed a known tree would be visible but instead we saw a new bench, or vice versa), the last two bullets may provide consistent DA situations. In case associations in planning and in inference are to the same (un)known variables we would have a consistent DA.

While different planning paradigms might diminish occurrences of inconsistent DA, e.g. by better predicting future associations, none can avoid it completely. Methods to better predict future observations/associations will be investigated in future work, potentially leveraging Reinforcement Learning (RL) techniques. As mentioned in Sect. 3.2, in this paper we do not predict occurrences of new landmarks, hence every new landmark in inference would result in inconsistent DA.

In the following section we provide a method to update inconsistent DA, regardless of a specific inconsistency scenario or a solution paradigm. This method utilizes the incremental methodologies of iSAM2 Kaess et al. (2012) in order to efficiently update the belief from the planning stage to have consistent DA with the succeeding inference.

3.5.2 Updating inconsistent DA

Inconsistent DA can be interpreted as disparate connections between variables. As discussed earlier, these connections, denoted as factors, manifest in rows of the Jacobian matrix or in factor nodes of a FG. Two FGs with different DA would thus have different graph topology. We demonstrate the inconsistent DA impact over graph topology using the example presented in Fig. 4: Fig. 4a represents the belief $b[X_{k+1|k}]$ from planning stage, and Fig. 4b represents the belief $b[X_{k+1|k+1}]$ from the inference stage. Even-though the same elimination order is used, the inconsistent DA would also create a different topology between the result-

Table 2 Notations for Sect. 3.5.2

Variable	Description
$\square_{t\|k}$	Of time t while current time is k
$\mathcal{FG}_{t\|k}$	Factor graph (FG) at time t
$\mathcal{T}_{t\|k}$	Bayes Tree (BT) at time t
$\mathcal{M}_{t\|k}$	Data Association (DA) at time t
$\mathcal{M}_t^{\bigcap}$	Consistent DA at time t
$\mathcal{M}_t^{rmv}$	DA at time t from planning inconsistent with inference, indicating factors to be removed
$\mathcal{M}_t^{add}$	DA at time t from inference inconsistent with planning, indicating factors to be added
$\{f_r\}_t^{rmv}$	Factors at time t from planning inconsistent with inference, to be removed
$\{f_s\}_t^{add}$	Factors at time t from inference inconsistent with planning, to be added
$\{X\}_t^{inv}$	All states at time t, involved in $\{f_r\}_t^{rmv}$ and $\{f_s\}_t^{add}$
$\mathcal{T}_t^{inv}$	Sub-BT of $\mathcal{T}_{t\|k}$ composed of all cliques containing $\{X\}_t^{inv}$
$\{X\}_t^{inv\star}$	All states at time t, related to the sub-BT $\mathcal{T}_t^{inv}$
$\mathcal{FG}_t^{inv}$	The detached part of $\mathcal{FG}_{t\|k}$ containing $\{X\}_t^{inv\star}$
$\mathcal{FG}_t^{upd}$	The FG $\mathcal{FG}_t^{inv}$ after DA update
$\mathcal{T}_t^{upd}$	The sub-BT eliminated from $\mathcal{FG}_t^{upd}$
$\mathcal{FG}_{t\|k}^{upd}$	The Factor Graph at time t with all-correct DA
$\mathcal{T}_{t\|k}^{upd}$	The Bayes Tree at time t with all-correct DA

ing BTs, e.g. the resulting BTs for the aforementioned FGs are Fig. 4d,e accordingly.

Performing action $u_{k|k+1}$, provides us with new measurements $z_{k+1|k+1}$, which are gathered to the factor set $\{f_j\}_{k+1|k+1}$ (see Appendix-B for factor definition). From the precursory planning stage we have the belief $b[X_{k+1|k}]$ along with the corresponding factor set $\{f_i\}_{k+1|k}$ for time $k+1$. Since we performed inference over this belief during the planning stage, we have already eliminated the FG, denoted as $\mathcal{FG}_{k+1|k}$, into a BT denoted as $\mathcal{T}_{k+1|k}$, e.g. see Fig. 4a,d, respectively.

We would like to update both the FG $\mathcal{FG}_{k+1|k}$ and the BT $\mathcal{T}_{k+1|k}$ from the planning stage, using the new factors $\{f_j\}_{k+1|k+1}$ from the inference stage. Without losing generality we use Fig. 4 to demonstrate and explain the DA update process. Let us consider all factors of time $k+1$ from both planning $\{f_i\}_{k+1|k}$ and inference $\{f_j\}_{k+1|k+1}$. We can divide these factors into three categories:

The first category contains factors with consistent DA - Good Factors. These factors originate from only the last two DA scenarios, in which both planning and inference considered either the same existing variable or a new one. Consistent DA factors do not require our attention (other than updating the measurements in the RHS vector). Indices of consistent DA factors can be obtained by intersecting the DA from planning with that of inference:

$$\mathcal{M}_{k+1}^{\bigcap} = \mathcal{M}_{k+1|k} \bigcap \mathcal{M}_{k+1|k+1}. \tag{64}$$

The second category - Wrong Factors, contains factors from planning stage with inconsistent DA to inference, which therefore should be removed from $\mathcal{FG}_{k+1|k}$. These factors can originate from all DA scenarios excluding the second. Indices of inconsistent DA factors from planning, can be obtained by calculating the relative complement of $\mathcal{M}_{k+1|k}$ with respect to $\mathcal{M}_{k+1|k+1}$:

$$\mathcal{M}_{k+1}^{rmv} = \mathcal{M}_{k+1|k} \setminus \mathcal{M}_{k+1|k+1}. \tag{65}$$

The third category - New Factors, contains factors from the inference stage with inconsistent DA to planning; hence, these factors should be added to $\mathcal{FG}_{k+1|k}$. These factors can originate from all DA scenarios excluding the first. Indices of inconsistent DA factors from inference, can be obtained by calculating the relative complement of $\mathcal{M}_{k+1|k+1}$ with respect to $\mathcal{M}_{k+1|k}$:

$$\mathcal{M}_{k+1}^{add} = \mathcal{M}_{k+1|k+1} \setminus \mathcal{M}_{k+1|k}. \tag{66}$$

We now use our example from Fig. 4 to illustrate these different categories:

- The first category - Good Factors, contains all factors from time $k+1$ that appear both in Fig. 4a,b, i.e. the motion model factor between x_k to x_{k+1}.
- The second category - Wrong Factors, contains all factors that appear only in Fig. 4a, i.e. the star marked factor in Fig. 4a. In this case the inconsistent DA is to an existing

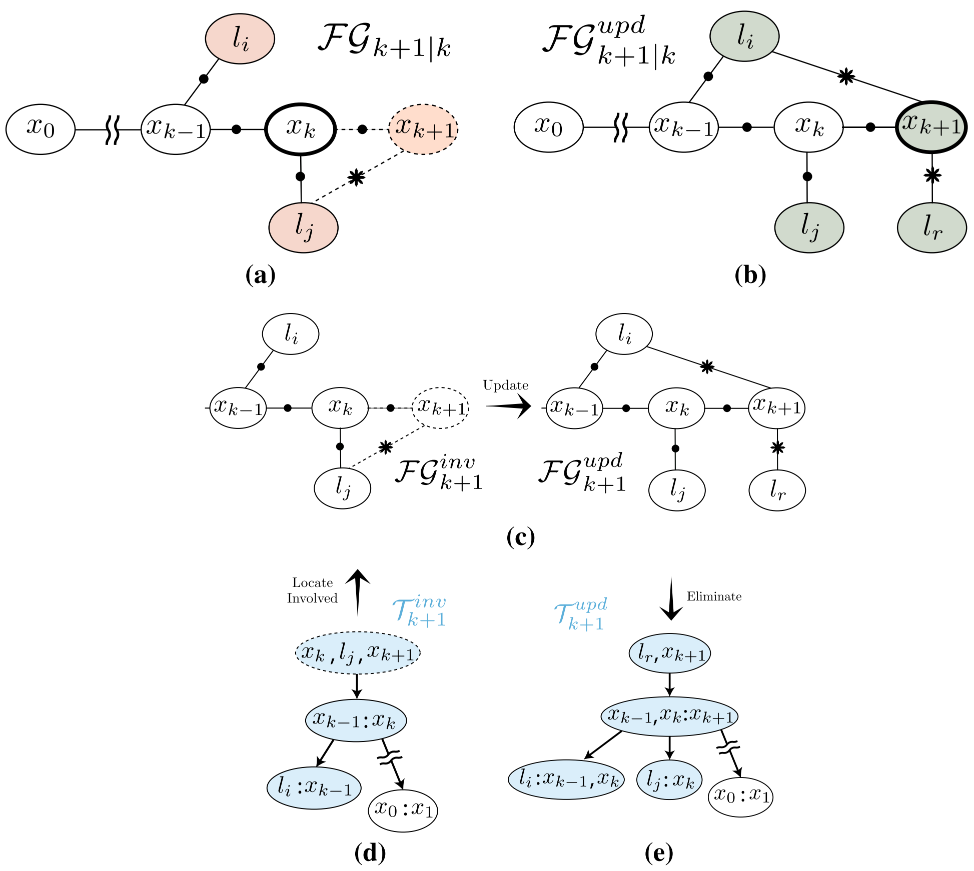


Fig. 4 The process of incremental DA update, following on `iSAM2` methodologies. **a** and **b** show factor graphs for $b[X_{k+1|k}]$ and $b[X_{k+1|k+1}]$, respectively, which differ due to incorrect association considered in the planning phase - l_j was predicted to be observed within planning, while in practice l_i and l_r were observed at time instant $k+1$. In **a**, current-time robot pose is bolded, horizon factors and states are dotted. Involved variables from DA comparison are marked in red in **a** and green in **b**. The belief $b[X_{k+1|k}]$, represented by a Bayes tree shown in **d**, is divided in two: sub Bayes tree containing all involved variables and parent cliques up to the root (marked in blue) and the rest of the Bayes tree in white. The former sub Bayes tree is re-eliminated by **i** forming the corresponding portion of the factor graph, as shown in the left figure of **c**; **ii** removing incorrect DA and adding correct DA factors, which yields the factor graph shown in the right figure of **c**; **iii** re-eliminating that factor graph into a sub Bayes tree, marked blue in **e**, and re-attaching the rest of the Bayes tree. While the obtained Bayes tree now has a correct DA, it is conditioned on (potentially) incorrect measurement values for consistent-DA factors, which therefore need to be updated (as detailed in Sect. 3.4), to recover the posterior belief $b[X_{k+1|k+1}]$

variable, landmark l_j was considered to be observed in planning but is not seen in the succeeding inference.

- The third category - New Factors, contains all factors that appear only in Fig. 4b, i.e. the star marked factors in Fig. 4b. In this case the inconsistent DA is both to an existing and a new variable. Instead of landmark l_j that was considered to be observed in planning, a different existing landmark l_i has been seen, along with a new landmark l_r.

Once the three aforementioned categories are determined, we use iSAM2 methodologies, presented in Kaess et al. (2012), to incrementally update $\mathcal{FG}_{k+1|k}$ and $\mathcal{T}_{k+1|k}$, see Alg. 1. The involved factors are denoted by all factors from planning needed to be removed (Wrong Factors), and all factors from inference needed to be added (New Factors),

$$\{f_r\}^{rmv}_{k+1} = \prod_{r\in\mathcal{M}^{rmv}_{k+1}} f_r \quad , \quad \{f_s\}^{add}_{k+1} = \prod_{s\in\mathcal{M}^{add}_{k+1}} f_s. \tag{67}$$

The involved variables, denoted by $\{X\}^{inv}_{k+1}$, are all variables related to the factor set $\{f_r\}^{rmv}_{k+1}$ and the factor set $\{f_s\}^{add}_{k+1}$ (Alg. 1, line 6), e.g. the colored variables in Figs. 4a,b accordingly. In $\mathcal{T}_{k+1|k}$, all cliques between the ones containing $\{X\}^{inv}_{k+1}$ up to the root are marked and denoted as the involved cliques, e.g. colored cliques in Fig. 4d. The involved cliques are detached and denoted by $\mathcal{T}^{inv}_{k+1} \subset \mathcal{T}_{k+1|k}$ (line 7). This sub-BT $\mathcal{T}^{inv}_{k+1}$, contains more variables than just $\{X\}^{inv}_{k+1}$. The involved variable set $\{X\}^{inv}_{k+1}$, is then updated to contain all variables from $\mathcal{T}^{inv}_{k+1}$ and denoted by $\{X\}^{inv\star}_{k+1}$ (line 8). The

Algorithm 1 - Data Association Update

```
1: function UPDATEDA($\mathcal{FG}_{k+1|k}$ , $\mathcal{M}_{k+1|k}$ , $\mathcal{FG}_{k+1|k+1}$ , $\mathcal{M}_{k+1|k+1}$)
2:     $\mathcal{M}^{rmv}_{k+1} \leftarrow \mathcal{M}_{k+1|k} \setminus \mathcal{M}_{k+1|k+1}$   ▷ indices of factors required to be removed
3:     $\mathcal{M}^{add}_{k+1} \leftarrow \mathcal{M}_{k+1|k+1} \setminus \mathcal{M}_{k+1|k}$   ▷ indices of factors required to be added
4:     $\{f_r\}^{rmv}_{k+1} \leftarrow \prod_{r \in \mathcal{M}^{rmv}_{k+1}} \{f_r\}_{k+1}$   ▷ factors required to be removed
5:     $\{f_s\}^{add}_{k+1} \leftarrow \prod_{s \in \mathcal{M}^{add}_{k+1}} \{f_s\}_{k+1}$   ▷ factors required to be added
6:     $\{X\}^{inv}_{k+1} \leftarrow Variables(\{f_r\}^{rmv}_{k+1}) \bigcup Variables(\{f_s\}^{add}_{k+1})$   ▷ get involved variables
7:     $\mathcal{T}^{inv}_{k+1} \leftarrow \mathcal{T}^{\{X\}^{inv}_{k+1}}_{k+1|k}$   ▷ get corresponding sub-BT
8:     $\{X\}^{inv\star}_{k+1} \xleftarrow{\text{get all variables}} \mathcal{T}^{inv}_{k+1}$   ▷ update involved variables
9:     $\mathcal{FG}^{inv}_{k+1} \leftarrow \mathcal{FG}^{\{X\}^{inv\star}_{k+1}}_{k+1|k}$   ▷ get corresponding sub-FG
10:    $\mathcal{FG}^{upd}_{k+1} \leftarrow [\mathcal{FG}^{inv}_{k+1} \setminus \{f_r\}^{rmv}_{k+1}] \bigcup \{f_s\}^{add}_{k+1}$   ▷ Update the sub Factor Graph
11:    $\mathcal{T}^{upd}_{k+1} \xleftarrow{\text{eliminate}} \mathcal{FG}^{upd}_{k+1}$   ▷ re-eliminate the updated sub-FG into BT
12:    $\mathcal{FG}^{upd}_{k+1|k} \leftarrow [\mathcal{FG}_{k+1|k} \setminus \mathcal{FG}^{inv}_{k+1}] \bigcup \mathcal{FG}^{upd}_{k+1}$   ▷ Update the Factor Graph
13:    $\mathcal{T}^{upd}_{k+1|k} \leftarrow [\mathcal{T}_{k+1|k} \setminus \mathcal{T}^{inv}_{k+1}] \bigcup \mathcal{T}^{upd}_{k+1}$   ▷ Update the Bayes Tree
14:    return $\mathcal{FG}^{upd}_{k+1|k}$ , $\mathcal{T}^{upd}_{k+1|k}$ .
15: end function
```

part of $\mathcal{FG}_{k+1|k}$, that contains all involved variables $\{X\}^{inv\star}_{k+1}$ is detached and denoted by $\mathcal{FG}^{inv}_{k+1}$ (line 9). While $\mathcal{T}^{inv}_{k+1}$ is the corresponding sub-BT to the acquired sub-FG $\mathcal{FG}^{inv}_{k+1}$.

In order to finish updating the DA, all that remains is updating the sub-FG $\mathcal{FG}^{inv}_{k+1}$ with the correct DA and re-eliminate it to get an updated BT. All factors $\{f_r\}^{rmv}_{k+1}$ are removed from $\mathcal{FG}^{inv}_{k+1}$, then all factors $\{f_r\}^{add}_{k+1}$ are added (line 10). The updated sub-FG is denoted by $\mathcal{FG}^{upd}_{k+1}$, e.g. update illustration in Fig. 4c.

By re-eliminating $\mathcal{FG}^{upd}_{k+1}$, a new updated BT, denoted by $\mathcal{T}^{upd}_{k+1}$, is obtained (line 11), e.g. the colored sub-BT in Fig. 4e. This BT is then re-attached back to $\mathcal{T}_{k+1|k}$ instead of $\mathcal{T}^{inv}_{k+1}$, subsequently the new BT is now with consistent DA and is denoted as $\mathcal{T}^{upd}_{k+1|k}$ (line 13). In a similar manner $\mathcal{FG}^{upd}_{k+1|k}$ is obtained by re-attaching $\mathcal{FG}^{upd}_{k+1}$ instead of $\mathcal{FG}^{inv}_{k+1}$ to $\mathcal{FG}_{k+1|k}$(line 12). At this point the DA in both the FG and the BT is fixed. For example, by completing the aforementioned steps, Figs. 4a,d will have the same topology as Figs. 4b,e.

After the DA update, the BT $\mathcal{T}^{upd}_{k+1|k}$ has consistent DA to that of $\mathcal{M}_{k+1|k+1}$. However, it is still not identical to $\mathcal{T}_{k+1|k+1}$ due to difference between measurement values predicted in planning to the values obtained in inference. The DA update dealt with inconsistent DA factors and their counterparts. For these factors the new measurements from inference were updated in the corresponding RHS vector values within the BT. The consistent DA factors, on the other hand, were left untouched; therefore, these factors do not contain the new measurement values from inference but measurement values from the planning stage instead. These inconsistent measurements are thus baked into the RHS vector $d_{k+1|k}$ and in the appropriate cliques of the BT $\mathcal{T}^{upd}_{k+1|k}$. In order to update the RHS vector $d_{k+1|k}$, or equivalently update the corresponding values within relevant cliques of the BT, one can use any of the methods presented in Sect. 3.4.

4 Results

In this section we present an extensive analysis of the proposed paradigm for `RUB inference` and benchmark it against the standard Bayesian inference approach using iSAM2 efficient methodologies as a proving-ground.

We consider the problem of autonomous navigation and mapping in an unknown environment as a testbed for the proposed paradigm, first in a simulated environment and later-on in a real-world environment (as discussed in the sequel). The robot performs inference to maintain a belief over its current and past poses and the observed landmarks thus far (i.e.full-SLAM), and uses this belief to decide its next actions within the framework of belief space planning. As mentioned earlier, our proposed paradigm is indifferent to a specific method of inference or decision making.

In order to test the computational effort, we compared inference update using iSAM2 efficient methodology, once based on the standard Bayesian inference paradigm Kaess et al. (2012) (here on denoted as `iSAM`), and second based on our proposed `RUB inference` paradigm.

As mentioned earlier, there are two changes required in a plan-act-infer system in order to facilitate the use of `RUB inference`. The first change involves the outputs of the planning stage, in addition to the standard output of the chosen (sub)optimal action sequence, we also output the future belief corresponding to said action. No other changes are required within the planning stage other than outputting more already available data. The second and more extensive change involves the inference stage, where we are required to incrementally update the aforementioned future belief with the newly acquired information.

All of our complementary methods (see Sect. 3.4), required to enable inference update based on the `RUB inference` paradigm, were implemented in MATLAB and are *encased within the inference block*. The `iSAM` approach uses the GTSAM C++ implementation with the supplied MATLAB wrapper Dellaert (2012). Considering the general rule of thumb, that MATLAB implementation is at least one order of magnitude slower, the comparison to iSAM as a reference is conservative. All runs were executed on the same Linux machine, with Xeon E3-1241v3 3.5 GHz processor with 32 GB of memory.

In order to get better understanding of the difference between our proposed paradigm and the standard Bayesian inference, we refer to the high-level algorithm diagram given in Fig. 1, which depicts a plan-act-infer framework. Fig. 1a represents a standard Bayesian inference, where the first and only inference update iteration is timed for comparison reasons. Fig. 1b shows our novel paradigm RUB inference, while the DA update, along with the first and only inference update iteration, are being timed for comparison. The computation time comparison is made only over the inference stage, since the rest of the plan-act-infer framework is *identical* in both cases.

As mentioned, our proposed paradigm does not affect estimation accuracy. We verify that in the following experiments, by comparing the estimation results obtained using our approach and iSAM. Both provide essentially the same results in all cases; we provide an explicit accuracy comparison with real-world data experiment (Sect. 4.2). It is worth mentioning that our paradigm is agnostic to the specific planning method or whether the action space is discrete or continuous. The solution procedure of a discrete action space revolves around solving the objective for multiple sets of action sequences, while under a continuous action space the candidate action sequence is continuously adjusted in the direction of the objective gradient up to some threshold. In both cases, the future belief corresponding to the selected (sub)optimal action will be calculated thus readily available as output.

All experiments consist of a 3D landmarks and 6D robot poses. The robot track is presented in a top view in order to allow the reader to examine the entire map with minimal occlusions.

4.1 Simulated environment

4.1.1 Basic analysis - sanity check

The purpose of this experiment is to provide with a basic comparison between the suggested paradigm for RUB inference and the existing standard Bayesian inference. This simulation performs a single horizon BSP calculation, followed by an inference step with a single inference update. The simulation provides a basic analysis of running time for each method, denoted by the *vertical* axis, for a *fully dense information matrix* and with no loop closures. The presented running time is a result of an average between 10^3 repetitions per step per method. Although a fully dense matrix does not represent a real-world scenario, it provides a sufficient initial comparison. The simulation analyzes the sensitivity of each method to the initial state vector size, denoted by the *horizontal* axis, and to the number of new factors, denoted by the different graphs. Since we perform a single horizon step with a single inference update, no re-linearization is necessary; hence, iSAM comparison is valid. The purpose of this check is to provide a simple sensitivity analysis of our methods to state dimension and number of new factors per step, while compared against standard batch update (denoted as STD) and iSAM paradigm. While both STD and iSAM are based on the standard Bayesian inference paradigm, the rest of the methods are based on the novel RUB inference paradigm.

Fig. 5 presents average timing results for all methods, while Figs. 5a–f represent different number of new rows added to the Jacobian matrix (equivalent to adding new measurements), [2 100 200 300 400 500] respectively. After inspecting the results, we found that for all methods, running time is a non-linear, positive-gradient function of the inference state vector size and a linear function of the number of new measurements. Moreover, the running time dependency over the number of new measurements diminish as the inference state vector size grows. For all inspected parameters our methods score the lowest running time with a difference of up to *three orders of magnitude* comparing to iSAM.

Figure 6 provides a zoom-in of Fig. 5, focusing on our suggested methods. Interestingly while we can clearly see that the OTM methodology is more efficient than the DU method, and the DU-OO is more efficient than DU, no such think can be said on OTM and OTM-OO. From inspecting Figs. 6a–f we can see that up to a state vector size of about 2500 there is no visible difference between OTM and OTM-OO performance, while for larger sizes the latter slightly outperforms the former.

Thus scoring all methods from the fastest to the slowest with a time difference of *four orders of magnitude* between the opposites:

OTM-OO ⇒ OTM ⇒ DU-OO ⇒ DU ⇒ iSAM ⇒ STD

4.1.2 BSP in unknown environment - consistent DA

The purpose of this experiment is to further examine the suggested paradigm of RUB inference, in a real world scenario, under the simplifying assumption of consistent DA. The second simulation performs BSP over continuous action space, in an unknown synthetic environment. In contrast to Sect. 4.1.1, since now the synthetic environment replicates a real world scenario, the obtained information matrix is now sparse (e.g. Fig. 16). A robot was given five targets (see Fig. 7a) while all landmarks were a-priori unknown, and was

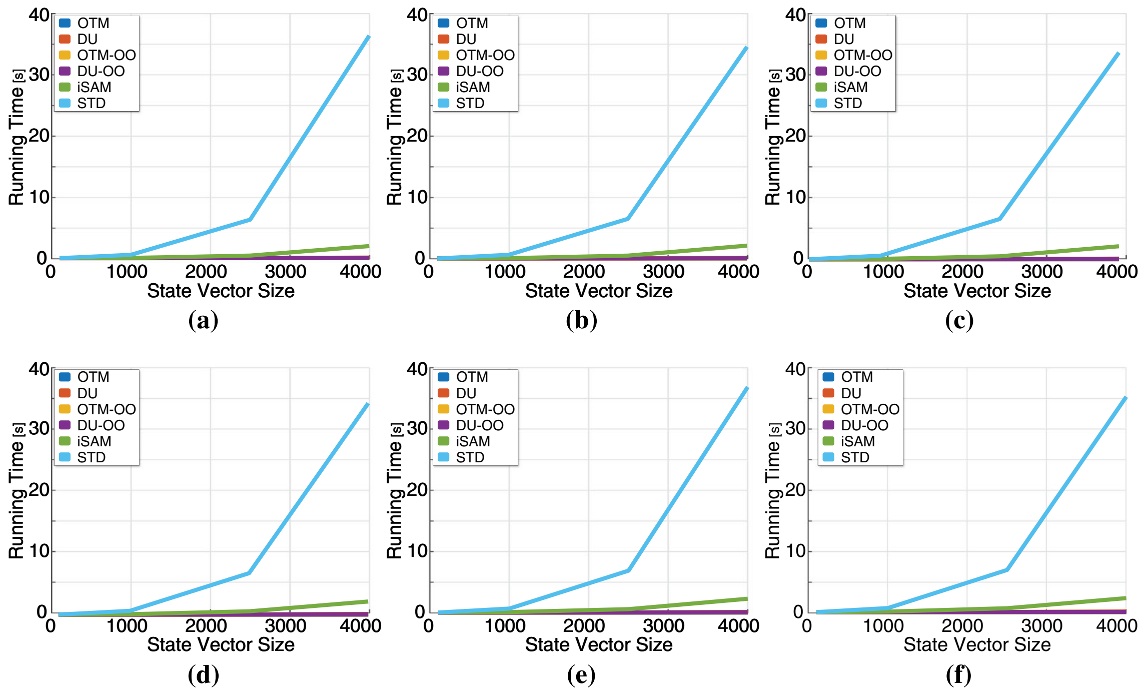


Fig. 5 Method comparison through basic analysis simulation, checking sensitivity to new added measurements and the size of the inference state vector over all the tested methods i.e. `STD`, `iSAM` and our four methods, i.e. `OTM`, `UD`, `OTM-OO` and `UD-OO`. Each graph represents a different number for new rows added to the Jacobian matrix **a** 2 rows **b** 100 rows **c** 200 rows **d** 300 rows **e** 400 rows **f** 500 rows. Due to orders of magnitude issues we also provide zoom-in to our four methods in Fig. 6

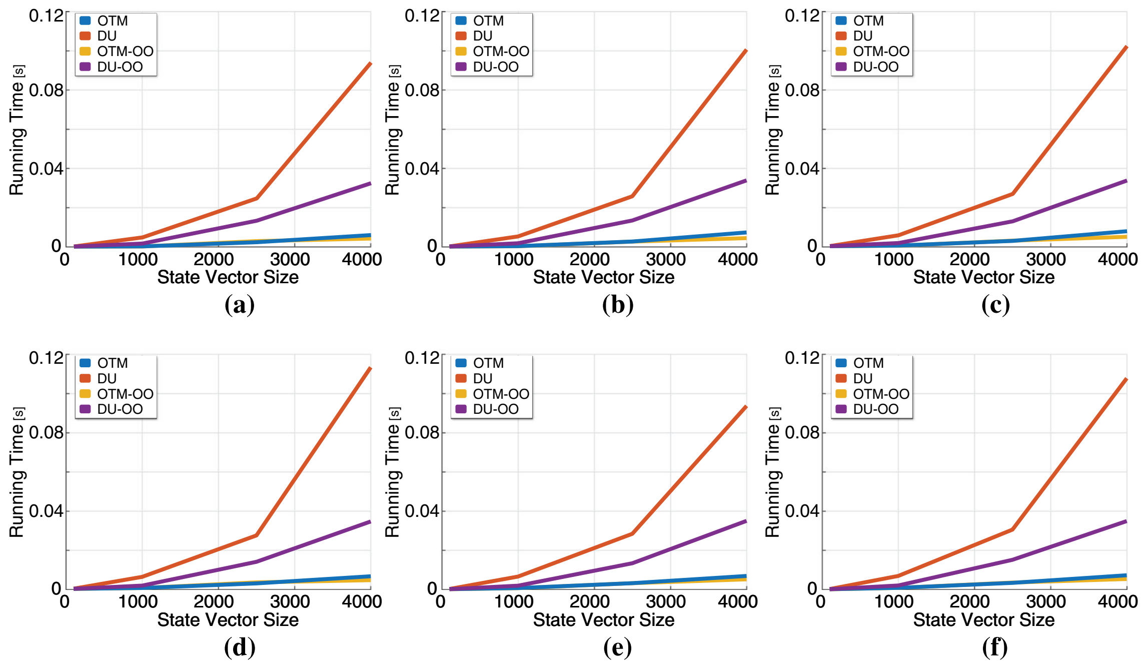
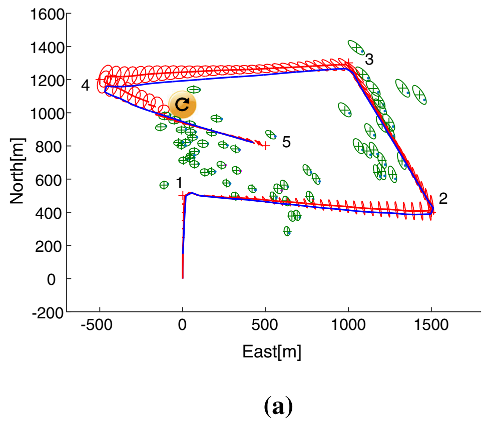


Fig. 6 Zoom-in on Fig. 5, checking sensitivity to new added measurements and the size of the inference state vector over our four methods i.e. `OTM`, `UD`, `OTM-OO` and `UD-OO`. Each graph represents a different number for new rows added to the Jacobian matrix **a** 2 rows **b** 100 rows **c** 200 rows **d** 300 rows **e** 400 rows **f** 500 rows



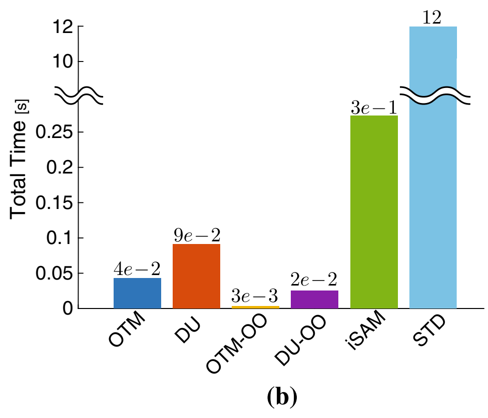


Fig. 7 Second simulation layout and results: **a** The Synthetic Environment, where landmarks are marked in green, targets are numbered and marked with red crosses, the ground truth is denoted by a blue line, the estimated trajectory is denoted by a red line while the covariance is visualized by red ellipse **b** Total average running time of inference update for each method

required to visit all targets whilst not crossing a covariance value threshold. The largest loop closure in the trajectory of the robot, and the first in a series of large loop closures, is denoted by a yellow ↻ sign across all relevant graphs. The robot performs BSP over continuous action space, with a finite horizon of five look ahead steps Indelman et al. (2015). During the inference update stage each of the aforementioned methods were timed performing the first inference update step. The presented running time is a result of an average between 10^3 repetitions per step per method. Similarly to Sect. 4.1.1, as can be seen in Fig. 7b, the suggested MATLAB implemented methods are up to *two orders of magnitude* faster than iSAM used in a MATLAB C++ wrapper. Interestingly, the use of sparse information matrices changed the methods' timing hierarchy. While OTM-OO still has the best timing results (3×10^{-3} s), *two orders of magnitude faster than* iSAM, OTM and DU-OO switched places. So the timing hierarchy from fastest to slowest is:

$$\text{OTM-OO} \Rightarrow \text{DU-OO} \Rightarrow \text{OTM} \Rightarrow \text{DU} \Rightarrow \text{iSAM} \Rightarrow \text{STD}$$

After demonstrating the use of our novel paradigm drastically reduce cumulative running time, we continue on to showing that in a few aspects it is also less sensitive. Figure 8 presents the performance results of each of the methods per simulation step. The upper graphs presents the number of new factors and new states per each step, while the lower graph presents the average running time of each method as a function of the simulation step. The ↻ sign, represents the first largest loop closure in a series of large loop closures. While some of the behavior presented in Fig. 8 can be related to machine noise, from carefully inspecting Fig. 8, alongside the trajectory of the robot in Fig. 7a, a few interesting observations can still be made. The first observation relates to the "flat line" area noticeable in the upper graph of Fig. 8b between time steps 60 – 90. This time steps range is equivalent to the path between the third and fourth targets, were the only factor added to the belief is motion based. As a result, a single new state (the new pose) is presented to the belief, along with a single motion factor. In this range, the timing results of iSAM DU and OTM present a linear behavior with a relatively small gradient. This gradient is attributed to the computational effort of introducing a single factor, containing a new state, to the belief. While the vertical difference between the aforementioned can be attributed to the sensitivity of each method to the number of states and factors in the belief.

From this observation, we can try to better understand the reason for the substantial time difference between the methods. Basing a method on RUB inference, rather than on standard Bayesian inference, will not magically change the computational impact of introducing factors or new states to the belief. However, because RUB inference is re-using calculations from precursory planning, the computational burden is being "paid" once, rather than twice as in the standard Bayesian inference. For the simple example of strictly motion propagation, since this motion based factor has already been introduced during precursory planning, under RUB inference it offers no additional computational burden. In the same manner, the reason RUB inference is less sensitive to the state dimensionality originates in calculations re-use. Under incremental update performed by iSAM, the state dimension is mostly noticeable when in need of re-ordering and/or re-eliminating states. Although same mechanisms also affect RUB inference, our method avoids them whenever they were adequately performed during the precursory planning, thus reducing inference computation time.

Another interesting observation refers to "pure" loop closures, were there are measurements with no addition of new variables to the state vector, i.e. measurements to previously observed landmarks. For the case of "pure" loop closures, STD, iSAM and the DU based methods (i.e. DU and DU-OO) experienced the largest timing spikes throughout the trajectory of each method while both OTM based methods experienced minor spikes if any.

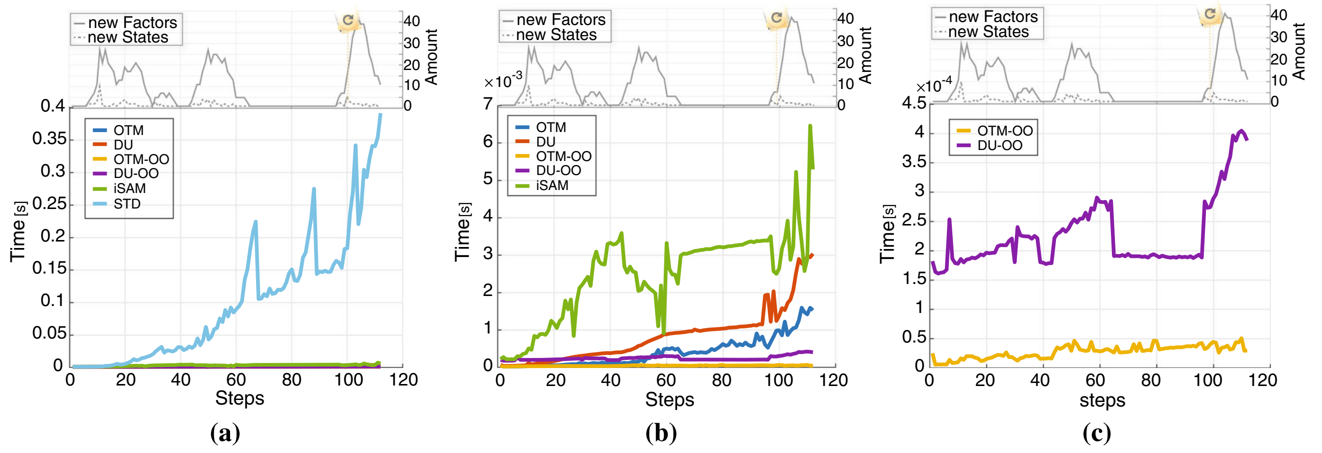


Fig. 8 Second simulation timing results for the scenario presented in Fig. 7a. Upper part of each graph provides indication on new factors and new states per computation step while the lower presents the methods timing results: **a** All six methods **b** OTM, DU, OTM-OO, DU-OOand iSAM methods **c** OTM-OO and DU-OO methods

By introducing the OO methodology to both DU and OTM, we drastically reduce the methods sensitivity to the motion propagation e.g. the once-positive gradient line in DU during time steps 60 − 90, turned into a flat line in DU-OO as can easily be seen in Fig. 8c. Moreover, while both DU and OTM present some sensitivity to different occurrences, i.e. the size of the state vector, new measurements and loop closures, this sensitivity is drastically reduced by introducing the OO methodology, e.g. OTM-OO is basically a flat line throughout the simulation as can easily be seen in Fig. 8.

In conclusion, our methods, based on RUB inference, particularly OTM-OO, seem to be more resilient to large loop closures that were already detected during planning, state vector size, belief size, number of newly added measurements or even the combination of the aforementioned.

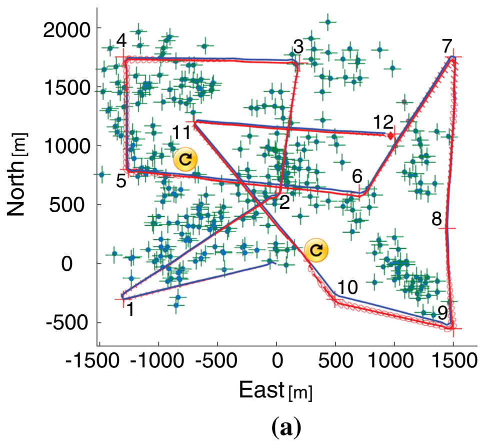
4.1.3 BSP in unknown environment - relaxing consistent DA assumption

The purpose of this experiment is to further examine the suggested paradigm of RUB inference, in a real world scenario, while relaxing the simplifying assumption of consistent DA. The third simulation performs BSP over continuous action space, in an unknown synthetic environment. A robot was given twelve targets (see Fig. 9a) while all landmarks were a-priori unknown, and was required to visit all targets whilst not crossing a covariance value threshold. The experiments presented in Sects. 4.1.1 and 4.1.2 were based on the simplifying assumption of consistent DA between inference and precursory planning, which can often be violated in real world scenarios. In this simulation we relax this restricting assumption and test our novel paradigm under the more general case where DA might be inconsistent.

The main reason for inconsistent data association lies in the perturbations caused by imperfect system and environment models. These perturbations increase the likelihood of inconsistent DA between inference and precursory planning. While the planning paradigm uses state estimation to decide on future associations, the further it is from the ground truth the more likely for inconsistent DA to be received. This imperfection is modeled by formulating uncertainty in all models (see Sect. 2).

For a more conservative comparison, in addition to the aforementioned, we force inconsistent DA between inference and precursory planning for all new variables. In contrast to planning paradigms that can provide DA to new variables, in addition to an unknown map, the robot's planning paradigm considers only previously-mapped landmarks. As a result of this limitation, the DA received from the planning stage can not offer new landmarks to the state vector. Consequently, each new landmark would essentially mean facing inconsistent DA, while the single scenario in which a consistent DA is obtained (see Sect. 3.5.1), occurs when both planning and inference are considering the same known landmark. Both perturbations caused by uncertainty and considering only previously mapped landmarks, resulted in just 50% DA consistency between planning and succeeding inference in this experiment. The remaining 50% contain factors related to both existing and new landmarks.

Following the findings of Sect. 4.1.2, out of the four suggested methods we choose to continue the comparison just with the OTM-OO method. While OTM-OO assumed consistent DA, the more general approach deals with inconsistent DA before updating the RHS vector. We denote the complete approach, updating DA followed by OTM-OO, as UD-OTM-OO, where UD stands for Update Data association.



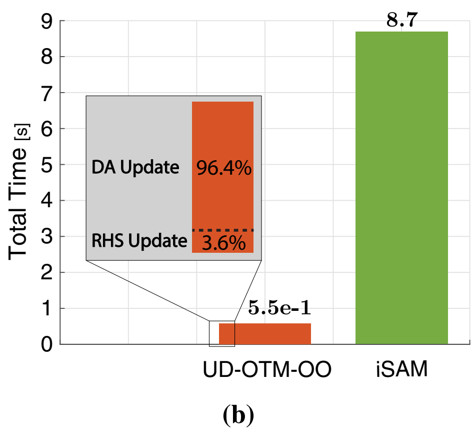


Fig. 9 Simulation layout and results: **a** The Synthetic Environment, where landmarks are marked in green, targets are numbered and marked with red crosses, the ground truth is denoted by a blue line, the estimated trajectory is denoted by a red line while the covariance is visualized by red ellipse. **b** Total average running time of inference update for each method, when 50% of the steps were with inconsistent DA

It is important to clarify that `UD-OTM-OO` and for consistent DA also `OTM-OO`, yield the same estimation accuracy as `iSAM`, since the inference update using `RUB inference` results in the same topological graph with the same values. Such comparison will be presented later on using a real-world data in Sect. 4.2. For that reason, the accuracy aspect will not be discussed further in this section. While the scenario presented in Fig. 9a contains at least ten large loop closures, for the readers convenience we marked two of them using yellow ⟳ signs. Same loop closures are also marked in Fig. 10 for comparison.

Figure 9b presents the cumulative computation time of the inference update phase throughout the simulation. We can see that the majority of `UD-OTM-OO` computation time, i.e. 96.4%, is dedicated to DA update while only 3.6% for updating the RHS vector. Although the need for DA update increased running time (as to be expected), `UD-OTM-OO` still outperforms `iSAM` by an order of magnitude.

In addition to the improvement in total computation time of the inference update stage, we continue on analyzing the "per step" behavior of `UD-OTM-OO`, and demonstrate that in a few aspects it is less sensitive than `iSAM`. Fig. 10a presents per step computation time of both `UD-OTM-OO` and `iSAM`, as well as the RHS update running time of `UD-OTM-OO`. Our suggested paradigm not only outperforms `iSAM` in the cumulative computation time, but also outperforms it for each individual step. While Fig. 10a presents the difference in average computation time per-step, Figs. f10b and f10c capture the reason for this difference as suggested in Sect. 4.1.2. Figure 10c presents the number of added factors in `iSAM` denoted by a green line, as opposed to in `UD-OTM-OO` denoted by an orange line, and the number of new variables per step denoted by a black line. Figure 10b presents the number of eliminations made during inference update in both methods. Number of eliminations reflects the number of involved variables in the process of converting FG into a BT (see Appendix-B and Algorithm 1 line 11 for the equivalent processes in `iSAM` and `UD-OTM-OO` accordingly).

After carefully inspecting both figures, alongside the robot's trajectory in Fig. 9a, the following observations can be made. Even with the limitation over the planning paradigm, both the number of new factors added and the number of re-eliminations during the inference update stage, are substantially smaller than their `iSAM` counterparts. These large differences are some of the reasons for `UD-OTM-OO`'s better performance. Due to the limitation over the planning paradigm, new observation factors (i.e. new landmarks added each step) in both `iSAM` and `UD-OTM-OO` are identical. While in `iSAM` new observation factors constitute a small fraction of total factors, for `UD-OTM-OO`, they constitute more than half of total factors. After comparing the re-elimination graph with the timing results for each of the methods, it appears both trends and peaks align, so we assume `UD-OTM-OO` as well as `iSAM` to be mostly sensitive to the amount of re-eliminations (further analysis is required).

Both re-elimination and added factors amounts, can be further reduced by smart reordering and relaxing the limitation over the planning paradigm accordingly.

As observed in Sect. 4.1.2, our method seems to be more resilient to loop closures. By inspecting the yellow ⟳ signs in Fig. 10c, we can see that in both cases, `iSAM` introduce around 50 factors of previously known variables (i.e. the black line representing new variables is zeroed), while `UD-OTM-OO` introduces no factors at all. These two loop closure examples beautifully demonstrate the advantage of using `RUB Inference`. For cases of consistent or partially consistent DA, when encountering a loop closure (i.e. observing a previously mapped landmark) our method saves valuable computation time since loop closures are only calculated once, in the planning stage (e.g. see timing response for loop closure at the appropriate yellow ⟳ signs in Fig. 10a).

Our method also seems to be less sensitive to state dimensionality. Inspecting steps $192 - 208$ and $263 - 275$ in Fig. 10c, we observe there are no new factors, i.e. the computation time is a result of motion factors; inspecting Fig. 10a we observe that in spite of the aforementioned,

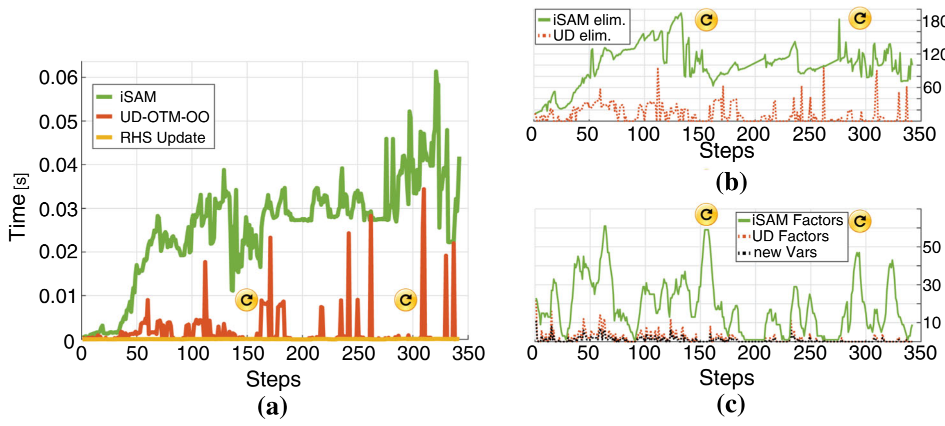


Fig. 10 Per-step analysis of the simulation presented in Fig. 7. In 50% of the steps, planning and succeeding inference are with consistent DA: **a** Per-step timing results of `iSAM` performing standard Beysian inference in green, `UD-OTM-OO` performing `RUB inference` in orange and the RHS update portion out of `UD-OTM-OO` in yellow. **b** Number of eliminations per-step, in the inference update stage for both `iSAM` and `UD-OTM-OO`. **c** Number of newly added factors in `iSAM` per step, newly added factors in `UD-OTM-OO` per step, and the number of new variables introduced to the belief per step

`iSAM` computation time is much larger than our method. From this comparison we can infer our suggested method is less sensitive to state dimensionality. As explained earlier, this originates in the reduced number of re-eliminations and state re-ordering in `RUB inference` when compared to `iSAM`, e.g. when the amount of re-eliminations in Fig. 10b is almost the same between the two (like in steps 171, 262, 310), the equivalent computation time in Fig. 10a is also almost identical.

4.2 Real-world experiment using KITTI dataset

After the promising performance in a simulated environment, we tested our paradigm for inference update via BSP in a real-world environment using KITTI dataset Geiger et al. (2013). The KITTI dataset, recorded in the city of Karlsruhe, contains stereo images, Laser scans and GPS data. For this work, we used the raw images of the left stereo camera, from the Residential category file: 2011_10_03_drive_0027, as measurements, as well as the supplied ground truth for comparison.

In this experiment we consider a robot, equipped with a single monocular camera, performing Active Full-SLAM in the previously unknown streets of Karlsruhe Germany. The robot starts with a prior over its initial pose and with no prior over the environment. At time k the robot executes BSP on the single step action sequence taken in the KITTI dataset at time $k+1$. At the end of each BSP session, the robot executes the chosen action, and receives measurements from the KITTI dataset. Inference update is then being performed in two separate approaches, the first following the standard Bayesian inference approach and the second following our proposed `RUB inference` approach. The inference update following each is compared for computation time and accuracy.

The following sections explain in-detail how planning and perception are being executed in this experiment.

4.2.1 Experiment parameters

For the readers convenience, this section covers all the parameters used for this experiment and were not provided by KITTI.

Prior belief standard deviation	$\begin{bmatrix} 1^o \cdot I_{3x3} & 0 \\ 0 & 1_{[m]} \cdot I_{3x3} \end{bmatrix}$
Motion Model standard deviation	$\begin{bmatrix} 0.5^o \cdot I_{3x3} & 0 \\ 0 & 0.5_{[m]} \cdot I_{3x3} \end{bmatrix}$
Observation Model standard deviation	$\begin{bmatrix} 1_{[px]} & 0 \\ 0 & 1_{[px]} \end{bmatrix}$
Camera Aperture	90^o
Camera acceptable Sensing Range	between $2_{[m]}$ and $40_{[m]}$

The motion and observation models that were used are (6) and (7) appropriately, where $h(.)$ is given by the pinhole camera model, and the zero mean Gaussian noise is stated above.

4.2.2 Planning using KITTI dataset

Our proposed approach for `RUB inference`, leverages calculations made in the precursory planning phase to update inference more efficiently. KITTI is a pre-recorded dataset with a single action sequence, i.e. the "future" actions of the robot are pre-determined. Nevertheless, we can still evaluate our approach by appropriately simulating the calculations that would be performed within BSP for that specific (and chosen) single action sequence. In other words, BSP involves belief propagation and objective function evaluations for different candidate actions, followed by identifying the best action via Eq. (12) and its execution.

In our case, the performed actions over time are readily available; hence, we only focus on the corresponding future beliefs for such actions given the partial information available to the robot at planning time. Specifically, at each time instant k, we construct the future belief $b[X_{k+1|k}]$ via Eq. (10) using the supplied visual odometry as motion model and future landmark observations. Future landmark observations are generated by considering only landmarks projected within the camera field of view using MAP estimates for landmark positions and camera pose from the propagated belief $b[X_{k+1|k}]$. As in this work the planning phase considers only the already-mapped landmarks, without reasoning about expected new landmarks, each new landmark observation in inference would essentially mean facing inconsistent DA.

To conclude, planning using the KITTI dataset is simulated over a single action in the following manner: current belief is propagated with future action, future measurements are generated by considering already-mapped landmarks, and future belief is solved. Since the "optimal" action is pre-determined by the KITTI dataset there is no need for an objective function evaluation.

4.2.3 Perception using KITTI dataset

After executing the next action, the robot receives a corresponding raw image from the KITTI dataset. The image is being processed through a standard vision pipeline, which produces features with corresponding descriptors Lowe (2004). Landmark triangulation is being made after the same feature has been observed at least twice, while following different standard conditions designed to filter outliers. Once a feature is triangulated, it is considered as a landmark, and is added as a new state to the belief. Note that the robot has access only to its current joint belief, consisting of the estimated landmark locations, and the robot past and present pose estimations. Once the observation factors (7) are added to the belief, the inference update is being made in two different and separate ways. The first, used for comparison, follows the standard Bayesian inference, by using the efficient methodologies of `iSAM2` in order to update inference. The belief of the preceding inference $b[X_{k|k}]$ is being updated with the new motion $\mathbb{P}(x_{k+1}|x_k, u_k)$ and observation factors $\prod_{j\in\mathcal{M}_{k+1|k+1}} \mathbb{P}(z^j_{k+1}|x_{k+1}, l_j)$, thus obtaining $b[X_{k+1|k+1}]$.

The second method follows our proposed paradigm for `RUB inference`. The belief from the preceding planning phase, $b[X_{k+1|k}]$, which corresponds to $u_{k|k+1}$ (see (26)), is updated with the new measurements. This update is done using `UD-OTM-OO` which consists of two stages, first using our DA update method (Sect. 3.5.2) which updates the predicted DA to the actual DA, followed by the `OTM-OO` method (Sect. 3.4.2) which updates measurement values.

4.2.4 Results - KITTI dataset

The robot travels 1400 steps in the unknown streets of Karlsruhe Germany, while relying only on a monocular camera for localization and mapping and without encountering any substantial loop closures. Differently than Sect. 4.1.2 and Sect. 4.1.3, where the landmarks were omnidirectional and therefore can be spotted from every angle as long as they were in sensing range, when using real world data the angle from which we see a landmark would have crucial affect on data association.Figure 11a presents the ground truth of the robot's trajectory in blue, the estimated robot's trajectory in dotted red and the estimated location of observed landmarks in green. Both `iSAM` and `UD-OTM-OO` produce the same estimation; therefore, the dotted red-line as well as the green marks represents both `iSAM` and `UD-OTM-OO` estimations.

Figure 11b presents the total computation time of inference update throughout the experiment, for both `iSAM`, and `UD-OTM-OO`. The importance of real-world data can be easily noticed by comparing Figs. 9b and 11b. While the RHS update portion of `UD-OTM-OO` secured its advantage of two orders of magnitude over `iSAM`, it is not the case with `UD-OTM-OO` as a whole. Although for real-world data, `UD-OTM-OO` is still faster than `iSAM`, the difference has decreased from order of magnitude in Fig. 9b, to less than half the computation time in Fig. 11b. Since the same machine has been used in both cases, the difference must originate from the data itself. As will be seen later in Fig. 13b, the number of measurements per step is substantially higher when using the real-world data, as well as the occurrences of inconsistent DA. It is worth stressing that `iSAM` implementation for inference update is C++ based, while `UD-OTM-OO` implementation consists of a mixture of MATLAB based and C++ based implementation, so under the use of the same platform the computation time difference is expected to be higher.

We continue by discussing the estimation difference, between `iSAM` and our method `UD-OTM-OO`. Although our method is algebraically equivalent to estimation via `iSAM`, for the reader's assurance we also provide estimation error comparison for both mean and covariance. Despite the algebraic equivalence, we expect to obtain small error values, related to numerical noise, which are different from absolute zero.

The estimation comparison results are presented in Fig. 12: the translation mean in Fig. 12a, the mean rotation of the robot in Fig. 12b and the corresponding covariances in Figs. 12c,d

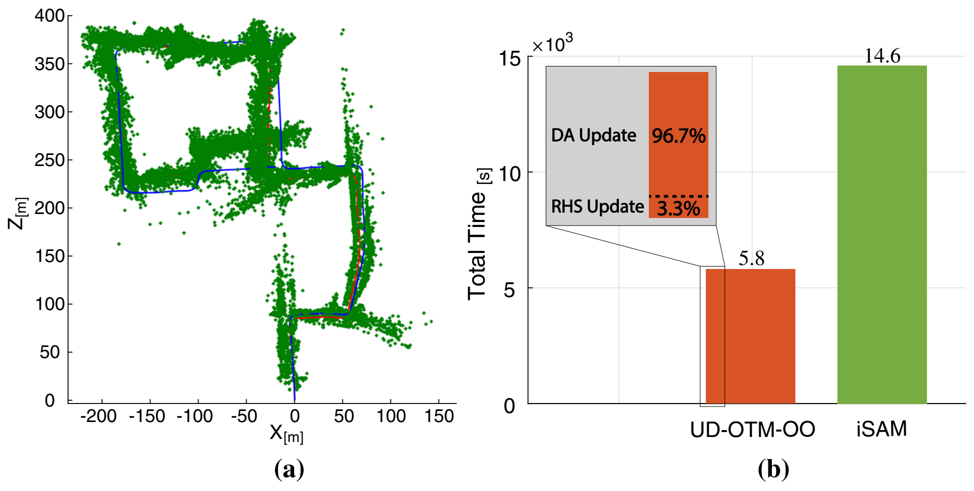


Fig. 11 Experiment layout and results: **a** The city of Karlsruhe, Germany, provided by the KITTI dataset. The robot ground truth is denoted in blue, the estimated trajectory denoted in dotted red line and the estimated landmark locations are denoted in green. **b** Total average running time of inference update for each method, when 100% of the steps were with inconsistent DA

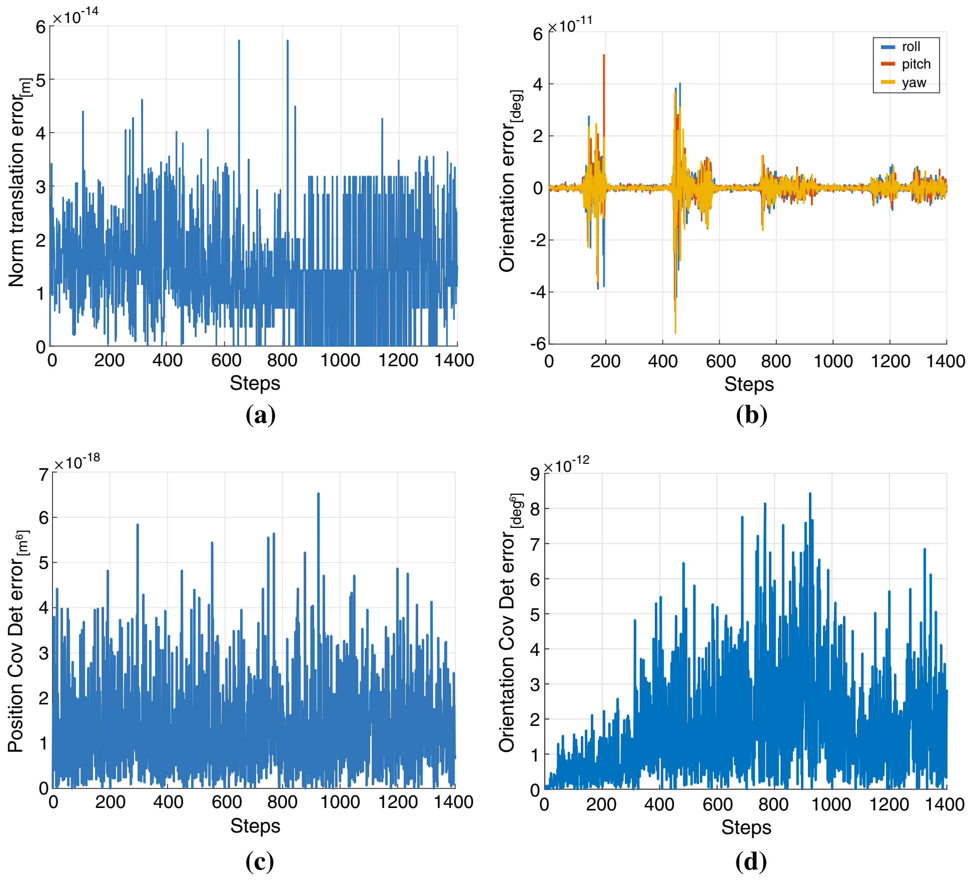


Fig. 12 Relative estimation error between `iSAM` and `UD-OTM-OO`, for KITTI dataset experiment **a** Relative translation error, calculated by taking the norm of the difference between the two translation vectors **b** Relative rotation error, calculated by taking the norm of the difference between the two orientation vectors, i.e. Euler angles **c** Relative position covariance error, calculated by taking the determinant of the difference between the two covariance matrices **d** Relative orientation covariance error, calculated by taking the determinant of the difference between the two covariance matrices

accordingly. The mean translation error is calculated by taking the norm of the difference between the two mean translation vectors. The mean rotation error is calculated by taking the norm of the difference between each of the mean body angles. The covariance error is calculated by taking the norm of the difference between the covariance determinants. As can be seen in Fig. 12, the error has a noise like behavior, with values of $10^{-14}_{[m]}$ for translation mean, $10^{-11}_{[deg]}$ for mean rotation angles, $10^{-3}_{[m]}$ for translation covariance and $10^{-2}_{[deg]}$ for rotation angles covariance. For all practical purposes, these values points to a negligible accuracy difference between the two methods.

Figure 13a presents the per-step computation time for inference update of `UD-OTM-OO` and `iSAM`, as well as the RHS update portion out of `UD-OTM-OO` for reference. The RHS Update timing of `UD-OTM-OO`, denoted by a yellow line, represents the per-step computation time of inference update through `RUB inference` for consistent DA,

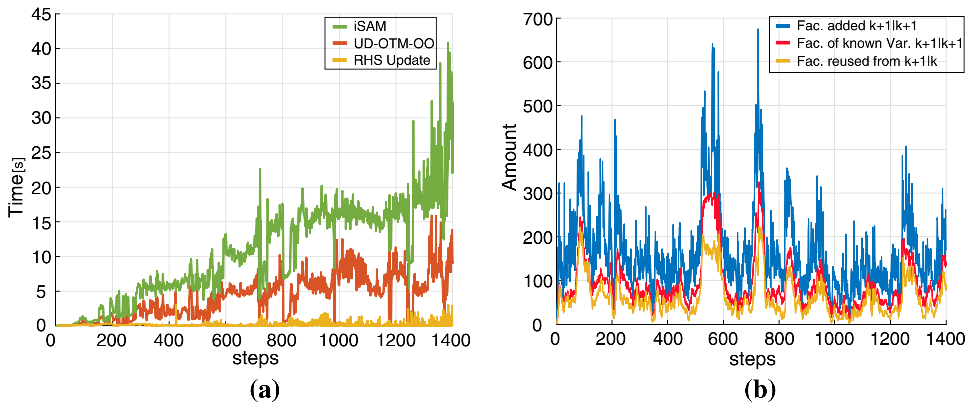


Fig. 13 Per-step analysis of computation time and added factors amount. **a** Inference update computation time per-step comparison between: `iSAM` - traditional Bayesian inference and `UD-OTM-OO` - inference update using belief from precursory planning. For reference the RHS update portion out of `UD-OTM-OO` is denoted in yellow. **b** Number of added factors per step. Number of all factors added in `iSAM` during inference at time $k+1$, denoted in blue. Number of factors added in `iSAM` during inference at time $k+1$ and relate to known variables, denoted in red. Number of factors that were originaly calculated during planning at time $k+1|k$ and were added by `UD-OTM-OO` in inference at time $k+1$, denoted in yellow

i.e. computation time for updating the RHS with the correct measurement values after the DA has been updated. `UD-OTM-OO` represents the per-step computation time of inference update through `RUB inference` for the entire process - DA update followed by RHS update. The difference in computational effort between the two, as seen in Fig. 13a, is equivalent to the computation time of the DA update, which represents the need to deal with inconsistent DA between belief from planning $b[X_{k+1|k}]$ and succeeding inference $b[X_{k+1|k+1}]$. The difference in computational effort between `UD-OTM-OO` and `iSAM` is attributed to the re-use of calculations made during the precursory planning. This calculation re-use manifests in salvaging factors that have already been considered during the precursory planning.

The reason for the considerable computational time differences between `UD-OTM-OO` and `iSAM` is better understood when comparing the factors involved in the computations of each method.

Figure 13b presents the sum of added factors per-step. In blue, the sum of all factors added at time $k+1|k+1$, as part of standard Bayesian inference update. In red, the portion of the aforementioned factors that relate to states which are already part of the belief $b[X_{k|k}]$. In yellow, the amount of factors added in time $k+1$ as part of `UD-OTM-OO` and are shared by both beliefs, $b[X_{k+1|k}]$ and $b[X_{k+1|k+1}]$, i.e. the amount of factors that were originally calculated in the precursory planning time, and were reused by `UD-OTM-OO`. It is worth stressing the noticeable difference between the number of measurements per step in Fig. 13b when compared to Fig. 10c. The former is exceeding the latter by an order of magnitude.

The difference between the yellow and blue lines in Fig. 13b represents the amount of factors “missing” from the belief $b[X_{k+1|k}]$ in order to match $b[X_{k+1|k+1}]$ (see Sect. 3.5.2), e.g. for step 725 only 142 have been reused (yellow line) while 675 were eventually added (blue line), leaving 533 new factors to be added during the DA update phase of `UD-OTM-OO`. This difference can be divided into factors containing only existing states and factors containing new states. Since the red line represents all factors of existing states, the difference between the red and blue lines represents all factors containing new states per time step, e.g. for step 725, out of the 675 factors added during inference (blue line), only 236 are related to previously known states (red line). As mentioned earlier in Sect. 4.2.2, in this experiment the prediction of future factors does not involve new states, apart from the next future pose(s). For that reason, the amount of factors added during planning has an upper bound represented by the red line, e.g. for step 725, the maximum number of factors that could have been utilized from precursory planning is 236 (red line). Future work can consider a prediction mechanism for new states, such work would set the upper bound somewhere between the red and blue lines.

The difference between the yellow and red lines, both related to factors of existing states, is attributed to the prediction accuracy of the planning stage. Since the factors represented by the red line are already part of the belief in planning time k, a perfect prediction mechanism would have added them all to the belief $b[X_{k+1|k}]$, e.g. for step 725, while there are 236 factors related to previously known states (red line), planning predicted only 142 of them (yellow line). Since the prediction is inherently imperfect (see Sect. 3.2), there would always be some difference between

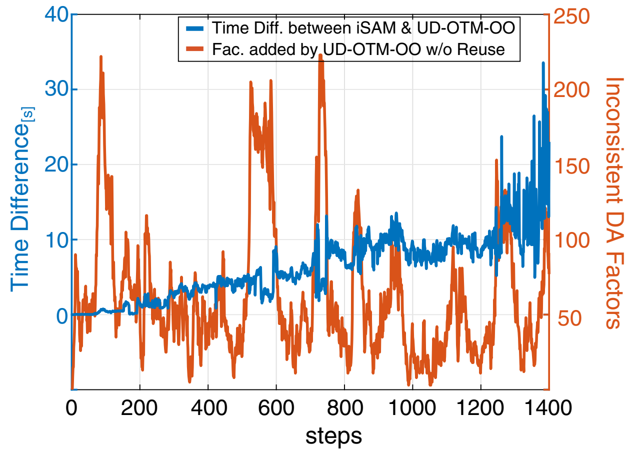


Fig. 14 Inference update computation time analysis between iSAM and UD-OTM-OO. The left vertical blue axis, represents the inference update computation time difference between iSAM and UD-OTM-OO, where positive values suggest $t_{\mathtt{iSAM}} > t_{\mathtt{UD\text{-}OTM\text{-}OO}}$. The right vertical orange axis, represents the number of factors per step added by UD-OTM-OO that were *not* reused from planning

the two. Reducing the gap between the red and yellow lines is a function of the prediction mechanism, while closing the gap further up to the blue line is a function of predicting new variables during planning.

After better understanding the meaning of Fig. 13b, comparing the two graphs in Fig. 13, reveals the connection between the added factors and the computation time, demonstrated by comparing steps 725 and 803 across the aforementioned. For time step 725, we have 675 new factors added in iSAM at inference, while only 142 factors that have been reused by UD-OTM-OO, this difference resulted in inference update running time of $4.2_{[s]}$ to UD-OTM-OO and $6.1_{[s]}$ to iSAM. For time step 803, we have 145 new factors added in iSAM at inference, while only 33 factors that have been reused by UD-OTM-OO, this difference resulted in inference update running time of $0.33_{[s]}$ to UD-OTM-OO and $6.9_{[s]}$ to iSAM. Although 6825 new landmarks were added to the state vector between steps 725 and 803 (calculated by the cumulative difference between the blue and red lines between steps 725 and 803), the time difference between UD-OTM-OO and iSAM increased, while UD-OTM-OO running time dropped. This increase in relative running time, in-spite of the substantial growth of the state vector, can be attributed to the drop in the number of factors needed to be added by the DA update phase of UD-OTM-OO.

As anticipated, the larger the gap between $b[X_{k+1|k}]$ and $b[X_{k+1|k+1}]$, i.e. more DA corrections to $b[X_{k+1|k}]$ are required in order to match $b[X_{k+1|k+1}]$, the smaller the computation time difference between RUB inference and standard Bayesian inference, as demonstrated in Fig. 14. The left vertical axis (denoted in blue) presents the computation time difference between iSAM and UD-OTM-OO such that positive values suggest $t_{\mathtt{iSAM}} > t_{\mathtt{UD\text{-}OTM\text{-}OO}}$. From this blue graph we notice that the time difference between iSAM and UD-OTM-OO is strictly positive and ascending up to fluctuations. While some of these fluctuations can be attributed to machine noise of the measurement process, we provide some explanation for the large time difference drops, i.e. the steps in which the computation difference between iSAM and UD-OTM-OO diminished. The number of factors added by UD-OTM-OO and were not reused from precursory planning are denoted by the orange line in Fig. 14. We can see correlation between large spikes in the number of factors added during the DA update phase of UD-OTM-OO (orange line), and the drops in the time difference between iSAM and UD-OTM-OO computation time difference (blue line), e.g. steps $554-591$ and $720-746$.

In contrast to previous experiments over synthetic data, we can better see here some dependency over the size of the belief in the UD-OTM-OO method. This dependency seems to be in correlation with that of iSAM2 although less intense, as can be seen by comparing the two methods in Fig. 13a. As in Fig. 10, we can attribute this correlation to the number of re-eliminations performed per step, which are a function of the newly added factors for iSAM and the DA update in UD-OTM-OO (see Sect. 3.5). As mentioned earlier, in each step UD-OTM-OO encounters inconsistent DA, judging by the difference between the blue and yellow lines in Fig. 13b, each step UD-OTM-OO deals with at least 100 factors that were not reused from planning. Since UD-OTM-OO makes use of iSAM2 methodologies in order to update inconsistent DA, as does iSAM2 to update inference, they share similar computational sensitivities, which manifest in similar computation time trends. This similarity sensitivity is attributed in our opinion to the elimination process required in order to introduce new factors into the belief. Future work for reducing eliminations by anticipating required ordering, would break this dependency and provide additional improvement in computation time as well as in reducing the sensitivity to state dimensionality.

5 Broader perspective

In this section we briefly discuss the motivation for RUB Inference, provide some broader perspective to possible future usage and discuss its usage outside iSAM2. As mentioned earlier, the RUB Inference paradigm deals with inference update within a plan-act-infer framework. By re-using calculations from the precursory planning session, it offers reduced computation time without affecting estimation accuracy.

Decision making under uncertainty in high dimensional state spaces is computationally intractable, and as such the majority of the plan-act-infer computation time can be ascribed to it. For example, let us consider BSP under the simplified Maximum Likelihood (ML) assumption, with a planning horizon of three lookahead steps and three candidate actions per step. The first level of the belief tree would consist of three future beliefs, one for each candidate action, each of which is propagated with each of the three candidate actions, resulting in nine future beliefs in the second level of the belief tree, and again for the last lookahead step with 27 future beliefs in the third and last level of the belief tree. This would result in total of 39 future beliefs that constitute the belief tree, i.e. 39 belief updates, whereas only a single belief update is required during inference update. In this toy example the computational load of inference update constitutes therefore only 2.5% of the plan-act-infer framework (assuming all belief updates have the same computational load). So why should we bother with the efficiency of the inference update process in the first place?

The answer to this question is twofold, the first part deals with `RUB Inference` paradigm as a stand-alone approach for inference update, and the second with its possible contribution to future research.

Although we present `RUB Inference` as part of a plan-act-infer system, it can also be used in the passive case, i.e. not as part of a plan-act-infer system. Imagine a set of candidate beliefs, calculated offline and stored away for future usage. When in-need to perform belief update, we can search this set of candidate beliefs for the belief closest to last posterior as well as to the newly received information. Once we locate this closest belief, we use `RUB Inference` to update it to match current information thus saving computational load without affecting accuracy. The reason we consider in this work `RUB Inference` as part of a plan-act-infer system, lies within the problem of locating the closest candidate belief. By using beliefs from precursory planning as candidates we have a small set of candidates to look through, moreover we can ensure that in the worst case scenario (i.e. all predictions from precursory planning are wrong) we would match Bayesian inference performance, thus averting from the complicated problem of searching within belief space. For the general case of having a set of previously calculated beliefs, used as candidates for re-use under `RUB Inference`, one would need to deal with few issues some of which are: how to store the beliefs to facilitate an efficient search, how to efficiently search the set of candidate beliefs, what high-dimensional belief-distance to use, how to interpret belief-distance into computational load i.e. what will be considered as close enough. It goes without saying that the computational load of locating the closest belief should be small enough for `RUB Inference` to still have a computational advantage.

Secondly, the paradigm shift suggested by `RUB Inference` provides a pathway to new and exciting research directions. For example, `RUB Inference` is a key building block in the new concept of Joint Inference and belief space Planning (`JIP`), first presented in Farhi and Indelman (2017) and later in Farhi and Indelman (2019b), which strives to create a unified framework that deals with both inference and BSP under the same governing system, thus allowing to maximize the calculation re-use potential available in both inference and planning.

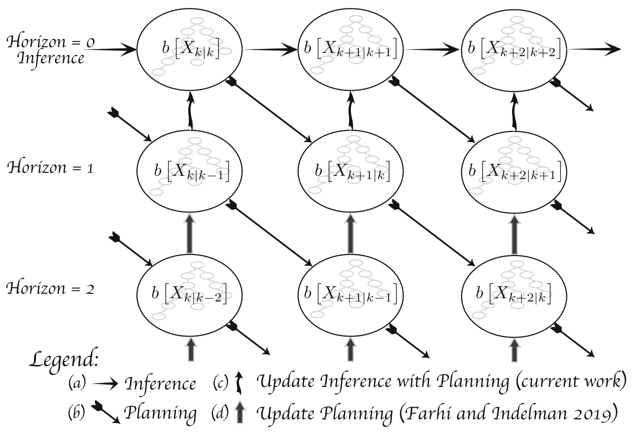


Fig. 15 Visualization of `JIP`, a novel approach to address both inference and belief space planning under a single process. Here $b[X_{k+1|k}]$ stands for the belief of the joint state in time instance $k+1$ while current time is k and each row stands for a different planning horizon. The relations between different beliefs in the graph are denoted by different arrows. **a** Inference; **b** Planning step; **c** Updating Inference with precursory planning (*this paper*); **d** Update planning with precursory planning

Fig. 15 provides a graphical illustration of `JIP`. The joint inference and belief space planning approach incorporates both inference and decision making stages into a single process. Each node in the graph represents a belief, i.e. $b[X_{k+1|k}]$ denotes the joint belief of state X at a future time instant $k+1$ given that the current time is k. The right facing arrows (a) denote inference at sequential time instances i.e. standard Bayesian inference. The diagonal arrows (b) represent optimal controls and future measurements that lead up to the appropriate beliefs. The upward facing arrow (c) represents our *current work* on `RUB Inference`, as it denotes inference update using precursory planning session. The upward facing arrow (d) represents our continued work named Incremental eXpectation BSP (`iX-BSP`) Farhi and Indelman (2019a), as it denotes updating a future belief using some previously calculated planning session. For the reader's convenience this `JIP` illustration is presented in 2D, but in-fact this 2D pattern is repeating itself in the 3D space to represents all possible future beliefs along with all possible candidate actions and future measurements, simply visualize rotating Fig. 15 around the top row representing inference.

While `RUB Inference` provides efficient inference update, denoted by (c) in Fig. 15, it is also relied upon to facilitate calculation re-use between different planning sessions, denoted by (d) in Fig. 15. As part of BSP, it is required to create a belief tree, as deep as the planning horizon, where each node in the said tree represents a future belief with specific candidate actions and future measurements. In Farhi and Indelman (2019a) we make use of `RUB Inference` paradigm in order to incrementally update the belief tree as part of planning, thus saving valuable computation time from the planning stage.

Consequently, apart from being a more computationally efficient approach for inference update, the paradigm shift suggested by `RUB Inference` provides a basis for exciting new research, i.e. `RUB Inference` is the building block that facilitates the `JIP` concept as a whole.

This entire paper is tightly entwined with iSAM2, yet we coin it as a general paradigm shift. How can it be used on different inference methods, if any? As stated in the Problem Statement (see Sect. 2.3), the paradigm shift suggested by `RUB Inference` is impervious to any specific paradigm used for inference or planning: The principle of `RUB Inference` (as laid out in Fig. 1b) involves updating a belief from a previous planning session as opposed to Bayesian inference where we update the last posterior belief. The motivation for this is the inherent similarities between inference and BSP (as seen in Sect. 3.3), which are not related to any specific inference method being used. In this work we chose to implement the principle suggested by `RUB Inference` over the `iSAM2` method, i.e. creating a version of `iSAM2` that follows `RUB Inference` rather than Bayesian inference. In order to implement `RUB Inference` on different inference methods, one must only implement the two building blocks (Fig. 1b) - updating DA, updating the actual measurements. While the presented implementation is in-fact `iSAM2` specific, it can be easily translated to other factor-based methods with little to no work. Different non-factor-based methods may require more adjusting, yet the paradigm of `RUB Inference` remains the same.

6 Conclusions

Conventional Bayesian inference updates the belief from a previous time step with new incoming information. In this work we introduced an alternative paradigm, utilizing the similarities between inference and planning to efficiently update inference using information from precursory planning phase. Given a future belief from precursory planning and newly acquired data, we appropriately update the former with the latter while taking into consideration data association inconsistencies which might occur. The resulting approach, `RUB inference`, saves valuable computation time in inference without affecting the estimation accuracy.

We evaluated our approach in simulation and using real-world data from the KITTI dataset, considering active SLAM as application, and compared it against `iSAM2`, a state-of-the-art incremental Bayesian inference approach. Results from real-world evaluation indicate that our approach is more efficient computationally by at least a factor of two compared to `iSAM2`, without affecting the solution accuracy. The improvement magnitude is in direct correlation with the quality of the prediction mechanism being used in planning, meaning a better prediction mechanism would increase the approach's efficiency. A particular appealing aspect of our method, that we demonstrated using synthetic data, is that loop closures computational burden during inference is elevated, thanks to the utilization of similar calculations already made during precursory planning. When loop closures were correctly predicted during the planning phase, our method utilized these calculations instead of re-calculating them in inference, resulting in reducing computation time by a factor of two orders of magnitude in the shown results.

This paper suggests a novel general concept for leveraging calculations from the decision making stage for efficient inference update, thus enabling to reduce inference computation time without affecting accuracy. Based on this concept, under the assumption of high-dimensional Gaussian beliefs, we developed approaches based on the square root information matrix, to efficiently update inference. We strongly believe this novel concept is applicable for more general distributions in any autonomous system involving both inference and decision making under uncertainty. Based on our findings, we strongly believe this paradigm shift opens new research directions and can be further extended in various ways, e.g. our ongoing work on `ix-BSP` - incremental expectation BSP Farhi and Indelman (2019a) leverages `RUB Inference` to reuse calculations across different planning sessions.

Appendix A: Derivation of equation (14)

In this appendix we complete the derivation of Eqs. (14) from (13). Let us consider the NLS presented in Eq. (13)

$$X^{\star}_{k|k} = \arg\min_{X_k} \|x_0 - x_0^{\star}\|^2_{\Sigma_0} + \sum_{i=1}^{k} \left[\|x_i - f(x_{i-1}, u_{i-1|k})\|^2_{\Sigma_w} + \sum_{j \in \mathcal{M}_{i|k}} \|z^j_{i|k} - h(x_i, l_j)\|^2_{\Sigma_v} \right].$$

In general, the motion model $f(\cdot)$ and the measurement model $h(\cdot)$ are non-linear functions. A standard way to solve this problem is the Gauss-Newton method, where a single iteration involves linearizing about the last known estimate, calculating the delta around this linearization point, and updating the latter with the former. This process should be repeated until convergence.

We start by linearizing the terms in (13) using first order Taylor approximation around the best estimate we have for the joined state $\bar{X}_{k|k-1}$ which is the state estimate for time k before including measurements, i.e. $X^*_{k|k-1}$.

The prior term yields,

$$x_0 - x_0^\star = \bar{x}_0 + \Delta x_0 - x_0^\star = \Delta x_0. \tag{68}$$

The motion model term yields,

$$x_i - f(x_{i-1}, u_{i-1|k}) = \bar{x}_i - f(\bar{x}_{i-1}, u_{i-1|k}) - \Sigma_w^{-\frac{1}{2}} \mathcal{F}_i \begin{bmatrix} \Delta x_{i-1} \\ \Delta x_i \end{bmatrix} \tag{69}$$

where $\Sigma_w^{-\frac{1}{2}} \mathcal{F}_i$ represents the Jacobian matrix of the motion model at time i, around the linearization point $\bar{x}_{i-1:i}$. The measurement model term yields,

$$z^j_{i|k} - h(x_i, l_j) = z^j_{i|k} - h(\bar{x}_i, \bar{l}_j) - \Sigma_v^{-\frac{1}{2}} \mathcal{H}_{i,j} \begin{bmatrix} \Delta x_i \\ \Delta l_j \end{bmatrix} \tag{70}$$

where $\Sigma_v^{-\frac{1}{2}} \mathcal{H}_{i,j}$ represents the jacobian matrix of the measurement model at time i around the linearization point $\left[\bar{x}_i, \bar{l}_j\right]^T$.

In order to re-write (13) into the common form of Least Squares $Ax = b$, we introduce Eqs. (68–70) back to (13),

$$\Delta X^\star_{k|k} = \underset{\Delta X_k}{\arg\min} \|\Sigma_0^{-\frac{1}{2}} \Delta x_0\|^2 + \sum_{i=1}^{k} \left[\|\Sigma_w^{-\frac{1}{2}} \Delta x_i - \mathcal{F}_i \Delta x_{i-1} - \breve{b}_i^{\mathcal{F}}\|^2 + \sum_{j \in \mathcal{M}_{i|k}} \|\mathcal{H}_{i,j} \begin{bmatrix} \Delta x_i \\ \Delta l_j \end{bmatrix} - \breve{b}_i^{\mathcal{H}}\|^2 \right],$$

where the RHS terms $\breve{b}_i^{\mathcal{F}}$ and $\breve{b}_i^{\mathcal{H}}$ are given by

$$\breve{b}_i^{\mathcal{F}} = \Sigma_w^{-\frac{1}{2}} \left(f(\bar{x}_{i-1}, u_{i-1|k}) - \bar{x}_i\right) \quad ,$$
$$\breve{b}_i^{\mathcal{H}} = \Sigma_v^{-\frac{1}{2}} \left(z^j_{i|k} - h(\bar{x}_i, \bar{l}_j)\right).$$

We now make use of the fact that the minimum sum of quadratic expressions is the minimum of each quadratic expression individually and is equal to zero. Thus enabling us to stack up all equations to form,

$$\Delta X^\star_{k|k} = \underset{\Delta X_k}{\arg\min} \|A_{k|k} \Delta X_k - b_{k|k}\|^2,$$

where the Jacobian matrix and the RHS are given by,

$$A_{k|k} = \begin{bmatrix} \Sigma_0^{-\frac{1}{2}} \\ \mathcal{F}_{1:k|k} \\ \mathcal{H}_{1:k|k} \end{bmatrix} \quad , \quad b_{k|k} = \begin{bmatrix} 0 \\ \breve{b}^{\mathcal{F}}_{1:k|k} \\ \breve{b}^{\mathcal{H}}_{1:k|k} \end{bmatrix}.$$

Appendix B: Inference as a graphical model

The inference problem can be naturally represented and efficiently solved using graphical models such as factor graph (FG) Kschischang et al. (2001) and Bayes tree (BT) Kaess et al. (2010). Since FG and BT graphical models pose key components in the suggested paradigm, the theoretical foundation is supplied next. We use Fig. 4 as illustration to belief representation in graphical models. Figure 4a and b are FG representations for the beliefs $b(X_{k+1|k})$ and $b(X_{k+1|k+1})$, respectively. BT representation of the belief is obtained through an elimination process, Fig. 4d presents the BT of $b[X_{k+1|k}]$ for the elimination order $x_0 \cdots l_i \rightarrow x_{k-1} \rightarrow x_k \rightarrow l_j \rightarrow x_{k+1}$, while Fig. 4e presents the BT of $b[X_{k+1|k+1}]$ for the elimination order $x_0 \cdots l_i \rightarrow x_{k-1} \rightarrow x_k \rightarrow l_j \rightarrow l_r \rightarrow x_{k+1}$.

A FG is a bipartite graph with two node types, factor nodes $\{f_i\}$ and variable nodes $\{\theta_j\} \in \Theta$. All nodes are connected through edges $\{e_{ij}\}$, which are always between factor nodes to variable nodes. A factor graph defines the factorization of a certain function $g(\Theta)$ as

$$g(\Theta) = \prod_i f_i(\Theta_i), \tag{71}$$

where Θ_i is the set of variables $\{\theta_j\}$ connected to the factor f_i through the set of edges $\{e_{ij}\}$. After substituting Θ with our joint state X and the factors $\{f_i\}$ with the conditional probabilities from Eq. (4) we receive the definition of the belief $b(X_{t|k})$ in a FG representation.

Through bipartite elimination game, a FG can be converted into a BN, this elimination is required for solving the Inference problem (as shown in Kaess et al. (2012)). After eliminating all variables the BN pdf can be defined by a product of conditional probabilities,

$$P(\Theta) = \prod_j P(\Theta_j | S_j), \tag{72}$$

where S_j is addressed as the *separator* of Θ_j, i.e. the set of variables that are directly connected to Θ_j. In order to ease optimization and marginalization, a BT can be used Kaess et al. (2012). By converting the BN to a directed tree, where the nodes represent *cliques* $\{C_r\}$, we receive a directed graphical model that encodes a factored pdf. Bayes Tree is defined using a conditional density per each node.

$$P(\Theta) = \prod_r P(F_r | S_r), \tag{73}$$

where S_r is the separator, defined by the intersection $C_r \bigcap \Pi_r$ of the clique C_r and the parent clique Π_r. The complement to the variables in the clique C_r is denoted as F_r, the *frontal variables*. Each clique is therefor written in the form

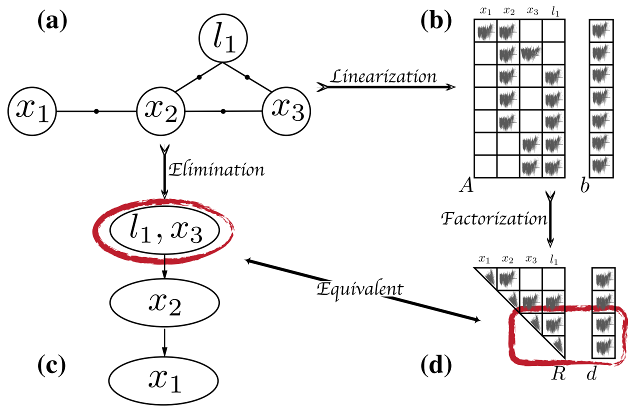


Fig. 16 The relations between different problem representations. **a** Factor graph **b** Jacobian matrix A with RHS vector b **c** Bayes Tree **d** Factorized Jacobian matrix R with equivalent RHS vector d

$C_r = F_r : S_r$.

The correspondence between matrix and graphical representation is conveniently demonstrated in Fig. 16. The first rows of R are equivalent to the deepest cliques in the BT, when the last rows of R equivalent to the root of the tree. The elimination order that created the BT is identical to the ordering of R state vector, and fill-ins in R equivalent to the connectivity of the corresponding BT.

Appendix C: Non-Zeros in Q matrix

In this appendix we discuss the number of non zeros in the rotation matrix $Q_{k+1|k+1}$, in order to do so we first cover the creation of $Q_{k+1|k+1}$, and later get to an expression for the number of non zeros and analyze it to gain better understanding over the governing parameters.

The rotation matrix $Q_{k+1|k+1}$ is created as part of the factorization of the Jacobian, designed to rotate the Jacobian into a square upper triangular form (e.g. Eqs. (17) and (25)). As such, we can deduce an expression for the number of non zeros in $Q_{k+1|k+1}$ as a function of the state size and the size of added factors, but first let us review how $Q_{k+1|k+1}$ is being created. Figure 17a illustrates a simple example for the Jacobian matrix $A^R_{k+1|k+1}$, where the precursory factorized Jacobian is denoted by $R_{k|k}$, the newly added factors by $\mathcal{A}_{k+1|k+1}$ and the columns denote the different states. The involved variables in $\mathcal{A}_{k+1|k+1}$ are marked with light blue and orange. As can be deduced from Fig. 17a, the number $A^R_{k+1|k+1}$ columns equals the joint state size at time $k+1$, and the number of $A^R_{k+1|k+1}$ rows equals the sum of the joint state size plus the number of $\mathcal{A}_{k+1|k+1}$ rows. The purpose of factorization is to rotate $A^R_{k+1|k+1}$ to a square upper triangular form without loosing information, i.e. so that ${A^R_{k+1|k+1}}^T A^R_{k+1|k+1} = {R_{k+1|k+1}}^T R_{k+1|k+1}$. While there are many different factorization algorithms, we would consider for simplicity without affecting generality the Given's Rotation (see Golub and Loan (1996)). Given's rotation creates $Q_{k+1|k+1}$ by a series of simple one cell rotations. For the simple case presented in Fig. 17, two rotations are required as presented in Fig. 17b. First the left-most non zero entry in $\mathcal{A}_{k+1|k+1}$, denoted by light blue, is addressed. The appropriate rotation matrix, consists of two off-diagonal non zeros denoted by dark blue, is denoted in Fig. 17b as the light blue $Q_{k+1|k+1}$. Next we are left to address the orange non zero entry in $\mathcal{A}_{k+1|k+1}$, while its appropriate rotation matrix, also consists of two off-diagonal non zeros denoted by dark red, is denoted in Fig. 17b as the orange $Q_{k+1|k+1}$. From Fig. 17b we can see that each sequential rotation matrix has the same number of non zeros, $diag\left(Q_{k+1|k+1}\right) + 2$, but due to the multiplication between them we get more non zeros as seen in Fig. 17c. For some intuition we marked the entries of the equivalent $Q_{k+1|k+1}$ presented in Fig. 17c, in accordance to the color coding in Fig. 17b.

Now that we understand that the number of non zeros in $Q_{k+1|k+1}$ is affected by the size of the joint state, the size of the newly added factors and the location of the left-most involved state, we are in position to formulate the expression for the number of non zeros in $Q_{k+1|k+1}$. We invite the reader to refresh his memory regarding the notations used in this analysis using Fig. 3, nevertheless all notations are also defined here.

Let j be the column index of the left-most involved state in the newly added factors $\mathcal{A}_{k+1|k+1}$, n^s be the size of the state vector (i.e. number of states multiplied by the state dimension), and n^f be the number of rows of $\mathcal{A}_{k+1|k+1}$ (i.e. number of factors multiplied by the factors' dimension). The number of non zeros can be defined as the sum of three values: the number of diagonal entries equal to 1, the contribution of the Jacobian line with the left-most state to the non zeros, and the contribution of the rest of the Jacobian lines. We will now calculate each of them.

As can be seen from Fig. 17b, the incremental rotation matrix (i.e. colored $Q_{k+1|k+1}$) created to rotate an entry in the i^{th} column, would have $i-1$ diagonal entries equal to 1. Since the left-most state is located in the j^{th} column the number of diagonal entries equal to 1 in $Q_{k+1|k+1}$ would be

$$j-1, \tag{74}$$

where j is bounded by the size of the state such that

$$j \in [1, n^s]. \tag{75}$$

The rotation matrix Q for rotating an entire Jacobian row located in the i^{th}, with a left-most non zero located in the j^{th} column, would have non zeros in the i^{th} row from column j up to the last column and an fully dense upper triangle of non

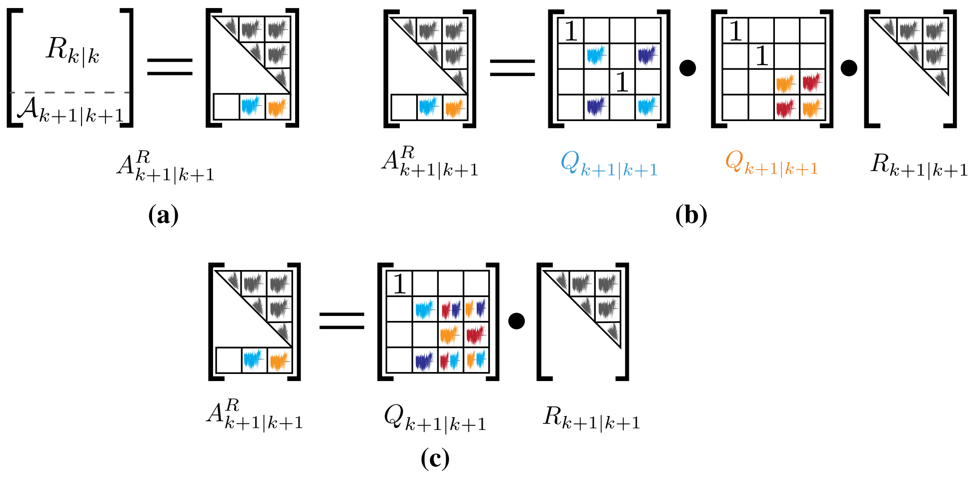


Fig. 17 **a** A Jacobian matrix at time $k+1$, consisting of the previously factorized Jacobian from time k and the linearized newly added factor from time $k+1$. The RHS visualize the non zeros of the aforementioned Jacobian. **b** Visualizing the factorization procedure of the Jacobian in **a** using two Given's rotation matrices. The light and dark colors represent the cosine and sine values respectively, attributed to each of the original non zeros in **a**. **c** Visualizing the non zeros in the rotation matrix required to factorize the Jacobian **a**, this rotation matrix is the product of the two matrices in **b**, as such the non zeros are affected by the cosine and sine values in **b**

zeros over the same columns. This means that rotating the row with the left-most index in column j would contribute the following number of non zeros to $Q_{k+1|k+1}$

$$\frac{n^2-n}{2}+n+n-1=\frac{n^2}{2}+\frac{3}{2}n-1, \tag{76}$$

where n is defined by

$$n=n^s+n^f-j+1. \tag{77}$$

Assuming the left-most state in the Jacobian is located in the j^{th} column, rotating the rest of the rows of the Jacobian will only add non zeros at the appropriate rows in Q, without adding new non zeros to the appropriate upper triangle. The remaining n^f-1 rows will contribute to $Q_{k+1|k+1}$ the following number of non zeros

$$\sum_{i=1}^{n^f-1}\left(n^s+n^f-i+1\right)=\left(n^f-1\right)\left(n^s+n^f+1\right)$$
$$-\frac{n^f\left(n^f-1\right)}{2}=(n^f-1)(n^s+\frac{n^f}{2}+1). \tag{78}$$

Evidently, the number of non zeros in $Q_{k+1|k+1}$ is given by

$$\underbrace{j-1}_{i}+\underbrace{\frac{n^2}{2}+\frac{3}{2}n-1}_{ii}+\underbrace{(n^f-1)(n^s+\frac{n^f}{2}+1)}_{iii}, \tag{79}$$

where term (i) in Eq. (79) denotes the number of diagonal entries equal 1, term (ii) in Eq. (79) denotes the non zeros added after factorizing the factor with the left-most state j, term (iii) in Eq. (79) denotes the non zeros added after factorizing the rest of the factors. It is worth stressing that the value of j is acutely affected by the ordering of the joint state vector. For better ordering, j would receive larger values.

Now that we have an expression to the number of non zeros in $Q_{k+1|k+1}$, we would like to investigate which part of it is dominant. In the sequel we reformulate Eq. (79) into a sum of quadratic terms, and then find conditions to determine which term is dominant.

We start by introducing (77) into Eq. (79) and using simple arithmetics in order to get quadratic forms,

$$\frac{1}{2}n^{s2}+n^{f2}+2n^sn^f+\frac{3}{2}n^s-n^sj+3n^f-n^fj+\frac{1}{2}j^2-\frac{3}{2}j-1 \tag{80}$$

$$\frac{1}{2}\left(n^s+n^f-j+\frac{3}{2}\right)^2+\frac{1}{2}n^{f2}+\frac{3}{2}n^f+n^sn^f-\frac{17}{8} \tag{81}$$

$$\underbrace{\frac{1}{2}\left(n^s+n^f-j+\frac{3}{2}\right)^2}_{a}+\underbrace{\frac{1}{2}\left(n^f+\frac{3}{2}\right)^2}_{b}+\underbrace{n^sn^f}_{c}-\frac{26}{8}. \tag{82}$$

We have three candidates to be the dominant part of Eq. (82), denoted by terms (a) (b) and (c). Let us examine them to decide which is the dominant one and under what conditions. First we can see that term (b) in (82) is a special case of term (a) in (82) where $j = n^s$. Subsequently we are left with comparing terms (a) and (c) in (82), i.e. we would like to check when

$$\left(n^s+n^f-j+\frac{3}{2}\right)^2>n^sn^f, \tag{83}$$

we define $a \triangleq n^s - j + \frac{3}{2}$ and get

$$a^2 + 2an^f + n^{f^2} - n^s n^f > 0. \tag{84}$$

So we can say term (a) in (82) is bigger than term (c) in (82) when

$$\left(n^s - j + \frac{3}{2} > \sqrt{n^s n^f} - n^f\right) \cup \left(n^s - j + \frac{3}{2} < -\sqrt{n^s n^f} - n^f\right). \tag{85}$$

Considering Eq. (75), we can dismiss $n^s - j + \frac{3}{2} < -\sqrt{n^s n^f} - n^f$ because the smallest the LHS can be is $\frac{3}{2}$, which will always be greater than the non positive number $-\sqrt{n^s n^f} - n^f$, so the condition on j so that term (a) is the dominant part of (82) is

$$n^s - \sqrt{n^s n^f} + n^f + \frac{3}{2} > j, \tag{86}$$

which after considering Eq. (75) is true if and only if

$$-\sqrt{n^s n^f} + n^f + \frac{3}{2} > 0. \tag{87}$$

We can now solve the aforementioned to get a condition to assure (86) holds,

$$n^f + \frac{3}{2} > \sqrt{n^s n^f} \tag{88}$$

$$n^{f^2} - (n^s - 3)f + \frac{9}{4} > 0 \tag{89}$$

$$\left(n^f > \frac{n^s - 3}{2} + \frac{\sqrt{n^{s2} - 6n^s}}{2}\right) \cup \left(0 < n^f < \frac{n^s - 3}{2} - \frac{\sqrt{n^{s2} - 6n^s}}{2}\right), \tag{90}$$

where $\frac{n^s-3}{2} - \frac{\sqrt{n^{s2}-6n^s}}{2}$ is non negative $\forall n^s$, and both conditions are defined for $n^s \geq 6$ which for a 6DOF problem means a single state. For a value of $n^s = 6$, $\frac{n^s-3}{2} - \frac{\sqrt{n^{s2}-6n^s}}{2} = 1.5$ and for $n^s = 7$, $\frac{n^s-3}{2} - \frac{\sqrt{n^{s2}-6n^s}}{2} < 1$ so affectively this condition is irrelevant $\forall n^s \neq 6$, so we are left with

$$\left(n^f > \frac{n^s - 3}{2} + \frac{\sqrt{n^{s2} - 6n^s}}{2}\right) \cup \left(n^s \geq 6\right). \tag{91}$$

Although this is the exact condition to insure term (a) is the dominant part of (82), in order to provide a more convenient condition we suggest an upper bound in the simple form of $n^f > n^s$.. Fig. 18 illustrates the effectiveness of the suggested bound in the form of the distance

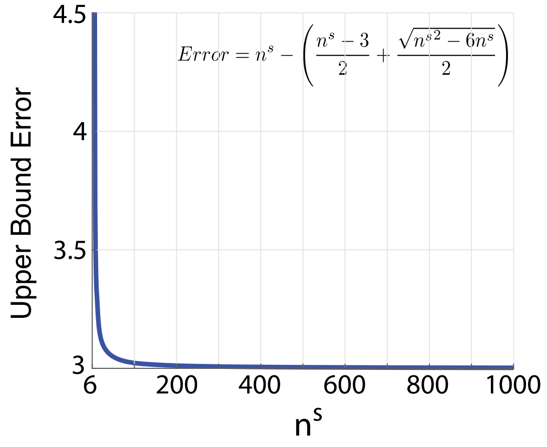


Fig. 18 Illustrating the effectiveness of the bound for $\frac{n^s-3}{2} + \frac{\sqrt{n^{s2}-6n^s}}{2}$ in the form of the error between the two as a function of different state sizes n^s

$$Error = n^s - \frac{n^s - 3}{2} + \frac{\sqrt{n^{s2} - 6n^s}}{2}. \tag{92}$$

For $n^s = 6$ the distance is 4.5, and for $n^s = 8$ it is already 3.5, which makes this bound very affective for simplicity reasons.

To conclude, term (a) is the dominant part of (82) if and only if the following holds

$$\left(n^f > \frac{n^s - 3}{2} + \frac{\sqrt{n^{s2} - 6n^s}}{2}\right) \cap \left(n^s \geq 6\right), \tag{93}$$

or for simpler upper bound

$$n^f > n^s \geq 6. \tag{94}$$

Otherwise, term (c) is the dominant part of (82), i.e. given simply the size of the state and the number of rows of the newly added factors we can determine what will be the governing expression for determining the number of non zeros in $Q_{k+1|k+1}$.

References

Bry, A., & Roy, N. (2011). Rapidly-exploring random belief trees for motion planning under uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 723–730).

Carlone, L., Censi, A., & Dellaert, F. (2014). Selecting good measurements via l1 relaxation: A convex approach for robust estimation over graphs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 2667–2674). IEEE.

Cunningham, A., Indelman, V., Dellaert, F. (May 2013). DDF-SAM 2.0: Consistent distributed smoothing and mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, Germany: Karlsruhe.

Davison, A., Reid, I., Molton, N., & Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM. *IEEE Transactions Pattern Analysis Machine Intelligent, 29*(6), 1052–1067.

Dellaert, F. (September 2012). Factor graphs and GTSAM: A hands-on introduction. Technical Report GT-RIM-CP&R-2012-002, Georgia Institute of Technology.

Dellaert, F., & Kaess, M. (2006). Square Root SAM: Simultaneous localization and mapping via square root information smoothing. *International Journal of Robotics Research, 25*(12), 1181–1203.

Eustice, R., Singh, H., & Leonard, J. (2006). Exactly sparse delayed-state filters for view-based SLAM. *IEEE Transactions Robotics, 22*(6), 1100–1114.

Farhi, E. I., & Indelman, V. (2017). Towards efficient inference update through planning via jip - joint inference and belief space planning. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Farhi, E. I., & Indelman, V. (May 2019a). ix-bsp: Belief space planning through incremental expectation. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Farhi, E. I., & Indelman, V. (May 2019b). Tear down that wall: Calculation reuse across inference and belief space planning. In *Toward Online Optimal Control of Dynamic Robots, Workshop in conjunction with IEEE International Conference on Robotics and Automation (ICRA)*.

Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*.

Golub, G., & Loan, C. V. (1996). *Matrix Computations* (3rd ed.). Baltimore: Johns Hopkins University Press.

Hartley, R. .I. ., & Zisserman, A. (2004). *Multiple View Geometry in Computer Vision* (2nd ed.). Cambridge University Press.

Haykin, S. S., et al. (2001). *Kalman filtering and neural networks*. Wiley Online Library.

Hollinger, G. A., & Sukhatme, G. S. (2014). Sampling-based robotic information gathering algorithms. *International Journal of Robotics Research, 33*, 1271–1287.

Indelman, V., Carlone, L., & Dellaert, F. (2015). Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments. *International Journal of Robotics Research, 34*(7), 849–882.

Indelman, V., Nelson, E., Dong, J., Michael, N., & Dellaert, F. (2016). Incremental distributed inference from arbitrary poses and unknown data association: Using collaborating robots to establish a common reference. *IEEE Control Systems Magazine (CSM), Special Issue on Distributed Control and Estimation for Robotic Vehicle Networks, 36*(2), 41–74.

Kaelbling, L. P., Littman, M. L., & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial intelligence, 101*(1), 99–134.

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

Kaess, M., Ila, V., Roberts, R., & Dellaert, F. (Jan 2010). The Bayes tree: Enabling incremental reordering and fluid relinearization for online mapping. *Technical Report MIT-CSAIL-TR-2010-021, Computer Science and Artificial Intelligence Laboratory*, MIT.

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

Kim, A., & Eustice, R. M. (2014). Active visual SLAM for robotic area coverage: Theory and experiment. *International Journal of Robotics Research, 34*(4–5), 457–475.

Kobilarov, M., Ta, D.-N., & Dellaert, F. (2015). Differential dynamic programming for optimal estimation. In: *IEEE International Conference on Robotics and Automation (ICRA)*, (pp. 863–869). IEEE.

Kschischang, F., Frey, B., & Loeliger, H.-A. (2001). Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory, 47*(2), 498–519.

Kurniawati, H., Hsu, D., & Lee, W. S. (2008). Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems (RSS)*, (vol. 2008).

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision, 60*(2), 91–110.

Olson, E., & Agarwal, P. (2013). Inference on networks of mixtures for robust robot mapping. *International Journal of Robotics Research, 32*(7), 826–840.

Papadimitriou, C., & Tsitsiklis, J. (1987). The complexity of markov decision processes. *Mathematics of operations research, 12*(3), 441–450.

Pathak, S., Thomas, A., & Indelman, V. (2018). A unified framework for data association aware robust belief space planning and perception. *International Journal of Robotics Research, 32*(2–3), 287–315.

Pineau, J., Gordon, G. J., & Thrun, S. (2006). Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research, 27*, 335–380.

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

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

Sunderhauf, N., & Protzel, P. (2012). Towards a robust back-end for pose graph slam. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 1254–1261). IEEE.

Ta, D.-N., Kobilarov, M., & Dellaert, F. (2014). A factor graph approach to estimation and model predictive control on unmanned aerial vehicles. In *International Conference on Unmanned Aircraft Systems (ICUAS)* (pp. 181–188). IEEE.

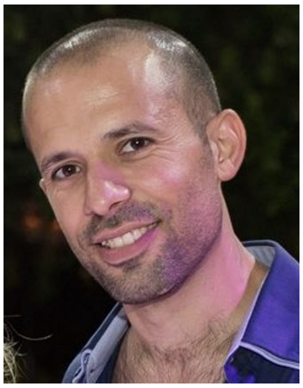
Thrun, S., Liu, Y., Koller, D., Ng, A., Ghahramani, Z., & Durrant-Whyte, H. (2004). Simultaneous localization and mapping with sparse extended information filters. *International Journal of Robotics Research, 23*(7–8), 693–716.

Todorov, E. (2008). General duality between optimal control and estimation. In *IEEE Conference on Decision and Control* (pp. 4286–4292). IEEE.

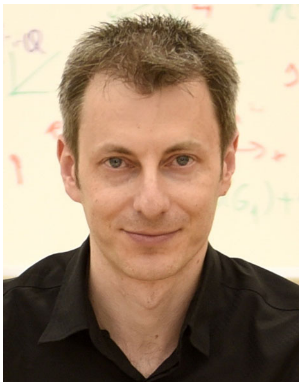
Toussaint, M. (2009). Robot trajectory optimization using approximate inference. In *International Conference on Machine Learning (ICML)*, (pp. 1049–1056). ACM.

Toussaint, M., & Storkey, A. (2006). Probabilistic inference for solving discrete and continuous state markov decision processes. In *International Conference on Machine Learning (ICML)*, (pp. 945–952). ACM.

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



Elad I. Farhi is a Ph.D. candidate at Technion Autonomous Systems Program (TASP), and a researcher at the Autonomous Navigation and Perception Lab at the Technion, Israel Institute of Technology. He holds a B.Sc. cum laude in Aerospace engineering, awarded by the Technion in 2013. His research focuses on joint incremental inference and belief space planning for online operations of autonomous systems.



Vadim Indelman is currently an Associate Professor in the Department of Aerospace Engineering at the Technion, and he is also a member of the Technion Autonomous Systems Program (TASP). Upon joining the Technion in July 2014 Dr. Indelman established the Autonomous Navigation and Perception Laboratory (ANPL), where he and his research group investigate problems related to single and multirobot collaborative autonomous navigation and perception, with a particular focus on online, accurate and reliable operation in uncertain and unknown environments. Prior to joining the Technion as a faculty member, Dr. Indelman was a postdoctoral fellow in the Institute of Robotics and Intelligent Machines (IRIM) at the Georgia Institute of Technology (between 2012 and 2014). He obtained his Ph.D. degree in Aerospace Engineering at the Technion - Israel Institute of Technology in 2011, and also holds B.A. and B.Sc. degrees in Computer Science and Aerospace Engineering, respectively, both awarded by the Technion in 2002.