Autonomous Robots (2024) 48:24
https://doi.org/10.1007/s10514-024-10180-6

Continuous planning for inertial-aided systems

Mitchell Usayiwevu[1] · Fouad Sukkar[1,2] · Chanyeol Yoo[1] · Robert Fitch[1] · Teresa Vidal-Calleja[1,2]

Received: 26 February 2024 / Accepted: 24 September 2024 / Published online: 12 October 2024

Abstract
Inertial-aided systems require continuous motion excitation among other reasons to characterize the measurement biases that will enable accurate integration required for localization frameworks. This paper proposes the use of informative path planning to find the best trajectory for minimizing the uncertainty of IMU biases and an adaptive traces method to guide the planner towards trajectories that aid convergence. The key contribution is a novel regression method based on Gaussian Process (GP) to enforce continuity and differentiability between waypoints from a variant of the RRT* planning algorithm. We employ linear operators applied to the GP kernel function to infer not only continuous position trajectories, but also velocities and accelerations. The use of linear functionals enable velocity and acceleration constraints given by the IMU measurements to be imposed on the position GP model. The results from both simulation and real-world experiments show that planning for IMU bias convergence helps minimize localization errors in state estimation frameworks.

Keywords Inertial-aided systems · Localization · IMU biases · Gaussian processes

1 Introduction

Localization is an important part of robotic navigation, in which the robot uses information from its onboard sensors to estimate its location over time (Bai et al., 2020; Panigrahi and Bisoy, 2021; Siegwart et al., 2011). This field of research has grown in popularity over the past two decades, with the majority of the localization approaches being split between optimization-based (Chang et al., 2016) or filtering-based (Bloesch et al., 2015).

Robust frameworks usually employ more than one sensing modality for localization (Cadena et al., 2016), with cameras, LiDARs and *Inertial Measurement Units* (IMUs) among the most commonly used sensors. The estimation algorithms used with any of these sensors need to account for sensor noises that corrupt the measurements. With IMUs however, there exists an additional layer of complexity introduced by the biases of the accelerometer and the gyroscope. These biases are embedded within the sensor measurements and need to be accounted for during state estimation in inertial-aided systems.

Whilst these IMU bias parameters can be calibrated offline, our focus is on developing robust autonomous systems suitable for long-term deployment, for example in remote locations. Given that sensor calibration can fluctuate due to wear and tear or sensor replacements, and considering that IMU bias parameters are particularly sensitive to variations in temperature and humidity, there is a compelling need for online self-calibration (Li et al., 2014; Yang et al., 2023). Most self-calibration works take a passive approach to handling the bias errors, where the estimation is carried out to find the robot location and bias with the same importance on each task (Perera et al., 2006).

In this work, we propose a more active approach of performing localization, in the context of inertial-aided systems. We use informative path planning (Popović et al., 2020), to find the best possible trajectory to estimate the IMU biases, thereby leading to more accurate localization estimates. This

✉ Mitchell Usayiwevu
Mitchell.Usayiwevu@alumni.uts.edu.au

Fouad Sukkar
fouad.sukkar@uts.edu.au

Chanyeol Yoo
chanyeol.yoo@uts.edu.au

Robert Fitch
robert.fitch@uts.edu.au

Teresa Vidal-Calleja
teresa.vidalcalleja@uts.edu.au

[1] UTS Robotics Institute, University of Technology Sydney, 81 Broadway, Ultimo, NSW 2007, Australia

[2] Australian Cobotics Centre, University of Technology Sydney, 81 Broadway, Ultimo, NSW 2007, Australia

extends our previous work (Usayiwevu et al., 2020), where we use Informative Path Planning to actively find trajectories that give the best possible extrinsic calibration for lidar-inertial systems.

Active planning is crucial in self-calibration for aided-inertial system because the quality of calibration parameter estimates depends on the trajectory that is used to collect the measurements (Hausman et al., 2017; Schneider et al., 2019). This means that different robot actions result in different estimation accuracies, with several basic motion profiles, known as degenerate motions, causing online sensor self-calibration failures (Li and Mourikis, 2014; Yang et al., 2020). It is evident from the extensive degeneracy analysis by Yang et al. (2019), Hausman et al. (2017) and Yang et al. (2023) that the IMU intrinsic calibration is sensitive to sensor motion and sufficient *excitation* to all 6 axes is necessary for accurate calibration parameters.

Our framework exploits an adaptive technique to guide the planning to minimize bias uncertainties first and localization uncertainty after convergence. This adaptive technique is used as the cost function in the RRT* variant that generates a set of discrete waypoints. However, to generate coherent IMU readings, the system requires continuous trajectories that are twice differentiable in position and once differentiable in orientation. In order to ensure such trajectories, Gaussian Process (GP) regression is used to interpolate the waypoints coming from our sampling-based planner. Linear operators are applied to the kernel function of the underlying GP on position in order to infer the first and second derivatives. Additionally, the use of linear functionals enable velocity and acceleration constraints to be added to the GP model as part of the measurement vector allowing for better control of not only the position trajectory but velocity and acceleration as well.

The key contribution is two-fold. (1) An informative path planning algorithm that adapts to prioritize convergence of IMU biases to improve localization accuracy. (2) A method to generate continuous and differentiable paths based on GP regression and linear operators, that allows embedding of constraints in the first and second derivatives (velocity and acceleration measurements) in the position trajectory.

Moreover, our algorithm is validated in a series of simulations and real-world experiments. The real-world experiments are carried out on a UR5 arm where our planner, which prioritizes IMU biases convergence adaptively, outperforms a joint optimization method.

2 Related work

There exists a number of works in the literature that have tackled continuous planning for IMU-aided systems. A common approach used, which is optimization based, generates minimum snap trajectories for quadrotor systems (Mellinger and Kumar, 2011). The approach generates trajectories that minimize the square of the norm of the fourth derivative of position (snap) using *Quadratic Program* (QP). Work in Mellinger et al. (2012), Burke et al. (2021) improve the numerical stability of the underlying QP and make it possible to have long-range trajectories composed of many segments for a finite time. Our regression method based on GPs can also deal with long-range trajectories and include as an addition their associated uncertainty.

Another common method of generating continuous trajectories is through parametric polynomial curves called Bezier curves. In Yassine et al. (2022), the authors compare the control performance of continuous trajectories generated by Bezier curves and B-splines. A major drawback of using Bezier curves is that they do not consider the dynamics of the system in the interpolation. In Hitz et al. (2017), authors use B-splines for continuous trajectories within an *Informative Path Planning* (IPP) framework to find an efficient path for maximizing information collected by the robot while respecting time and energy constraints (Wakulicz et al., 2022). Similarly, authors in Bähnemann et al. (2017), Usayiwevu (2022) and Usayiwevu et al. (2023) employ sampling-based motion planning to explore the space of possible beliefs and find a maximally informative trajectory within a user-defined budget to reduce model parameter uncertainty. Although this is similar to our work, in that we use an IPP for maximizing information to reduce localization uncertainty, we prioritize IMU bias convergence in order to improve the quality of the bias estimation and ultimately improve state estimation.

GPs are used in motion planning in Le Gentil and Vidal-Calleja (2023), Mukadam et al. (2018) and Marchant and Ramos (2014). In Mukadam et al. (2018), the GP representation is tightly coupled to the gradient descent-based optimization algorithm for full state estimation. Our proposed framework uses GP regression in a loosely coupled manner that allows the planner to focus on reducing bias uncertainty. Authors in Marchant and Ramos (2014) use GP regression for space-time modeling then apply Bayesian Optimization to estimate the best path to collect new observations in an exploratory manner. Unlike their work, we use GP regression to interpolate between sample points from our planner that finds the best trajectory to improve localization accuracy. Additionally, the application of linear operators to the kernel function of underlying GP in position in our method allows inference of the first and second derivatives.

Inertial-based active localization, active SLAM, navigation and exploration can be found in Qin et al. (2019), Elisha and Indelman (2017) and Papachristos et al. (2017). These works operate by providing approaches in which the action the robot takes and the measurements collected are the most efficient for reducing localization uncertainty. Authors of Liu et al. (2005) improve localization accuracy for inertial-aided

systems by reducing the noise level in the raw acceleration measurements.

A range of works use active planning to find trajectories that can be used to estimate the most optimal parameters in robotic systems. For instance, Hausman et al. (2017) proposes a framework for finding a trajectory that will provide an optimal convergence of self-calibration parameters of a nonlinear system. The framework minimizes an observability-aware cost function, that can be used jointly with other optimization objectives. This framework however assumes GPS availability, which is not considered in our setup.

Webb et al. (2014) introduces a method for planning optimal learning control policies in real time using an approximate POMDP solver. These policies identify controls that are approximately optimal for learning one or more unknown parameters associated with the model of a robot, while jointly respecting any other objectives encoded in the cost functions. Murali et al. (2016) use active planning in belief space for autonomous extrinsic calibration of an exteroceptive sensor. Planning is performed such that it actively reduces calibration parameter uncertainty at each instant in a previously unknown environment. Similarly, authors in Maye et al. (2016) use active calibration for the identification of extrinsic system parameters.

While active planning is considered in Maye et al. (2016) and Murali et al. (2016), they only consider extrinsic calibration and not IMU intrinsics as in our work. The closest work to ours is presented in Elisha and Indelman (2017), where the authors propose an active calibration framework of the intrinsic parameters such as IMU bias. Their method allows the robot to select the best path that improves its own state estimation, based on belief space planning.

The primary distinction of our work and the joint optimization frameworks presented in Hausman et al. (2017), Webb et al. (2014), Murali et al. (2016), Elisha and Indelman (2017), Maye et al. (2016) lies in the adaptive nature of our cost function. Unlike the static objectives in the cost functions of the aforementioned works, our framework dynamically adjusts the cost function to minimize the IMU biases trace before the convergence of IMU bias estimates, and minimize the robot position trace after IMU bias estimates convergence. This ensures quicker convergence of IMU bias uncertainties, which ultimately leads to better localization accuracy.

3 Problem statement and overview

3.1 Inertial based systems

Consider an inertial-aided system whose state can be estimated by any probabilistic estimation framework, and the state is modelled as a multivariate Gaussian distribution $\mathcal{N}(\mathbf{x}, \mathbf{P})$. The mean state defined as $\mathbf{x} = (\mathbf{r}, \mathbf{v}, \mathbf{R}, \mathbf{b}_f, \mathbf{b}_w, \mathbf{c}, \mathbf{z})$, where $\mathbf{r}$ is the position of the IMU in the world frame W, $\mathbf{v}$ is the velocity of the IMU in W, $\mathbf{R} \in SO(3)$ is the orientation (represented as a rotation matrix from W to the IMU frame I), $\mathbf{b}_f$ is additive bias on accelerometer, $\mathbf{b}_w$ is additive bias on gyroscope, $\mathbf{c}$ and $\mathbf{z}$ represent the translation and rotation components of the extrinsic calibration between the IMU and exteroceptive sensor used, and $\mathbf{P}$ is the state covariance matrix.

The process model and measurement model of the additional exteroceptive sensor are defined as

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\epsilon}(t)) \tag{1}$$

$$\mathbf{y}(t) = h(\mathbf{x}(t), \boldsymbol{\upsilon}(t)), \tag{2}$$

where $\mathbf{u}(t)$ is the control input, the process noise is $\boldsymbol{\epsilon}(t) \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\epsilon(t))$ and the measurement noise is $\boldsymbol{\upsilon}(t) \sim \mathcal{N}(0, \boldsymbol{\Sigma}_\upsilon(t))$.

The IMU provides linear acceleration $\tilde{\mathbf{f}}(\mathbf{t}_i)$ and angular velocity measurements $\tilde{\boldsymbol{\omega}}(\mathbf{t}_i)$ at time $\mathbf{t}_i$ with $i = (1, ..., T)$ in the inertial reference frame. The linear acceleration of the IMU in W is denoted as $\mathbf{f}_W$, while $\boldsymbol{\omega}$ is the angular velocity of the IMU frame relative to W. The relationship between the IMU measurements and $\mathbf{f}_W(\mathbf{t}_i)$ and $\boldsymbol{\omega}(\mathbf{t}_i)$ is given by,

$$\tilde{\mathbf{f}}(t) = \mathbf{R}_W^t(t)^\top(\mathbf{f}_W(t) - \mathbf{g}) + \mathbf{b}_f(t) + \boldsymbol{\eta}_f(t) \tag{3}$$

$$\tilde{\boldsymbol{\omega}}(t) = \boldsymbol{\omega}(t) + \mathbf{b}_\omega(t) + \boldsymbol{\eta}_{\boldsymbol{\omega}}(t)\,, \tag{4}$$

where $\mathbf{g}$ is the gravity vector in W, and $\boldsymbol{\eta}_f$ and $\boldsymbol{\eta}_\omega$ are zero-mean Gaussian sensor noises with covariance matrix Σ_{η_f} and Σ_{η_ω} for the linear accelerations and angular velocities respectively.

At time t given an IMU, the system kinematics $f(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\epsilon}(t))$ is given by:

$$\dot{\mathbf{R}}_W^t(t) = \mathbf{R}_W^t(t)(\tilde{\boldsymbol{\omega}}(t) - \mathbf{b}_\omega(t) - \boldsymbol{\eta}_\omega(t))^\wedge \tag{5}$$

$$\dot{\mathbf{v}}_W^t(t) = \mathbf{R}_W^t(t)(\tilde{\mathbf{f}}(t) - \mathbf{b}_f(t) - \boldsymbol{\eta}_f(t)) + \mathbf{g} \tag{6}$$

$$\dot{\mathbf{r}}_W^t(t) = \mathbf{v}_W^t(t)\,, \tag{7}$$

and the IMU sensor biases modelled by a Brownian motion,

$$\dot{\mathbf{b}}_f(t) = \boldsymbol{\eta}_{\mathbf{b}_f}(t) \tag{8}$$

$$\dot{\mathbf{b}}_\omega(t) = \boldsymbol{\eta}_{\mathbf{b}_\omega}(t)\,, \tag{9}$$

where $\boldsymbol{\eta}_{\mathbf{b}_f}$ and $\boldsymbol{\eta}_{\mathbf{b}_\omega}$ are zero-mean Gaussian noise of the accelerometer and gyroscope biases, with variances given by Σ_{b_f} and Σ_{b_ω} respectively. Thus, the control input is given by,

$$\mathbf{u}(t) = \begin{bmatrix} \tilde{\mathbf{f}}(t) - \boldsymbol{\eta}_f(t) \\ \tilde{\boldsymbol{\omega}}(t) - \boldsymbol{\eta}_{\boldsymbol{\omega}}(t) \end{bmatrix} \quad \boldsymbol{\epsilon}(t) = \begin{bmatrix} \boldsymbol{\eta}_{\mathbf{b}_f}(t) \\ \boldsymbol{\eta}_{\mathbf{b}_\omega}(t) \end{bmatrix}. \tag{10}$$

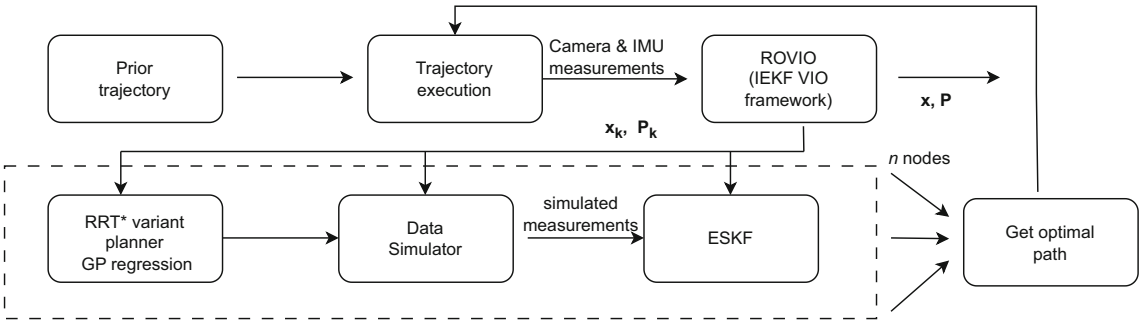


Fig. 1 Overview of the proposed framework

Note the symbol $\wedge$ is the skew-symmetric matrix operator.

3.2 Problem formulation

Given an inertial-aided system moving in an unknown environment and an associated estimation framework, the aim is to find the continuous optimal trajectory $\boldsymbol{\pi}^*$ of the system, in the space of all trajectories ψ for maximum gain in the information-theoretic measure,

$$\boldsymbol{\pi}^* = \underset{\pi \in \psi}{\operatorname{argmax}} \frac{\mathrm{I}[\mathrm{M}(\pi)]}{\mathrm{T}(\pi)}, \quad \text{s.t.} \quad C(\pi) \leq B, \tag{11}$$

where $\mathrm{I} : \mathbb{R}^6 \mapsto \mathbb{R}$ is the utility function that evaluates the information gain in localization. The function $\mathrm{M} : SE(3) \mapsto \mathbb{R}^6$ obtains discrete sensor measurements along the trajectory $\boldsymbol{\pi}$ with $\mathrm{T} : SE(3) \mapsto \mathbb{R}$ as corresponding travel time. The cost of the path $\mathrm{C} : SE(3) \mapsto \mathbb{R}$ given by the planner cannot exceed a predefined budget B. The utility function above is formulated to compute the expected reduction in IMU biases uncertainty and robot localization uncertainty.

3.3 Overview

We propose an Informative Path Planning framework as described in Sect. 3.2 that directly takes into account the impact of the biases $\mathbf{b}_f$ and $\mathbf{b}_\omega$ embedded in the IMU measurements $\tilde{\mathbf{f}}$ and $\tilde{\boldsymbol{\omega}}$ to maximize localization information gain or in other words minimize localization uncertainty. Consider a visual-inertial system carrying out a localization task in the environment $\mathcal{E}$ using a probabilistic estimation framework. Given the camera and IMU measurements acquired during the execution of a path $\pi_{0:k}$ between times $t = 0$ and $t = t_k$, the state $\mathbf{x}_k$ and covariance matrix $\mathbf{P}_k$ of the system at time $t = t_k$ are estimated with a VIO framework ROVIO (Bloesch et al., 2015) which is based on the IEKF.

Figure 1 presents an overview of the method as a block diagram. First, a pre-defined path $\pi_{0:k}$ is executed and ROVIO provides the corresponding state $\mathbf{x}_k$ and covariance $\mathbf{P}_k$ estimates. The last state of the prior trajectory $\mathbf{x}_k$ is set as the start node for the planning algorithm. A variant of the RRT* planner incrementally builds a tree of candidate trajectories by sampling in the linear position and orientation space. Because of the constraints that the IMU imposes on the system, we require smooth and continuous trajectories that are differentiable at least 2 times, to ensure smooth and continuous linear position, velocity and acceleration trajectories, and angular positions and velocity trajectories. We propose here our tailor-made interpolation method based on GP regression to interpolate the waypoints from the sampling based planner. The continuous trajectory and estimated map are used for simulating inertial measurements in the data simulator block, which are then used to propagate the joint full state expected covariance $\mathbf{P}_k$ into the posterior $\mathbf{P}_{k+1}$ in the ESKF block.

This posterior joint expected covariance matrix is marginalized and the block corresponding with either the localization or biases estimate is used to determine information content in a particular candidate trajectory. For selecting which block to marginalize, an adaptive technique is employed. It selects the biases uncertainty to guide the planning before convergence of the bias uncertainties and the localization uncertainty is selected after convergence. Computation of the information content in a trajectory is based on the A-optimality criterion.

The planner considers poses that have the most *excitation* in the acceleration and angular velocity space which helps the IMU biases to converge more quickly and produce more accurate localization estimates. Note that our planner can work with any filtered-based inertial-aided estimation framework as we will show in the experiments section using two existing frameworks. The estimation framework to be used only needs to estimate the state, feature map and their associated covariance matrix which are then used in our Informative Path Planning framework for decision-making.

4 GPs for continuous trajectories

A *Gaussian Process* (GP) can be seen as a distribution over functions in the continuous domain, such that every finite collection of the random variables has a multivariate normal distribution, Williams and Rasmussen (2006). It is completely specified by its mean function $\mu(\boldsymbol{t})$ and covariance function $k(\boldsymbol{t}, \boldsymbol{t'})$ for a real process $\xi : \mathbb{R}^d \mapsto \mathbb{R}$ where d is the dimension of the input.

$$\mu(\mathbf{t}) = \mathbb{E}[\xi(\mathbf{t})] \tag{12}$$

$$k(\mathbf{t}, \mathbf{t}') = \mathbb{E}[(\xi(\mathbf{t}) - \mu(\mathbf{t}))(\xi(\mathbf{t}') - \mu(\mathbf{t}'))]. \tag{13}$$

In this work, we consider a zero-mean GP defined over time $\mathbf{t}$, thus $d = 1$, which is used to generate continuous linear position, velocity and acceleration trajectories, angular positions and velocities used in a state estimation framework (see Fig. 2). Our GP is defined as,

$$\xi(\mathbf{t}) \sim \mathcal{GP}(\mathbf{0}, k(\mathbf{t}, \mathbf{t}')) \tag{14}$$

$$\boldsymbol{\gamma}_i = \xi(\mathbf{t}_i) + \Omega_i, \tag{15}$$

where $\boldsymbol{\gamma}_i, \Omega_i \in \mathbb{R}$ and the joint covariance of the noise $\boldsymbol{\Omega} = (\Omega_1, \Omega_2, ...\Omega_n)$ is assumed to be given by the matrix $\Sigma_{\mathbf{e}}$. Given a sequence of waypoints in position $\boldsymbol{\gamma} = (\boldsymbol{\gamma_1}, \boldsymbol{\gamma_2}, \ldots, \boldsymbol{\gamma_n})$, the joint distribution of the observed position waypoints and the continuous position values at the test locations can be written as,

$$\begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\xi}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(\mathbf{t}, \mathbf{t}) + \Sigma_{\boldsymbol{\Omega}} & K(\mathbf{t}, \mathbf{t}_*) \\ K(\mathbf{t}_*, \mathbf{t}) & K(\mathbf{t}_*, \mathbf{t}_*) \end{bmatrix}\right). \tag{16}$$

The position posterior mean and covariance are given by the predictive Gaussian process regression as,

$$\begin{aligned} \bar{\boldsymbol{\xi}}_* &= \mathbb{E}[\boldsymbol{\xi_*}|\mathbf{t}, \boldsymbol{\gamma}, \mathbf{t}_*] \\ &= K(\mathbf{t}_*, \mathbf{t})[K(\mathbf{t}, \mathbf{t}) + \Sigma_{\boldsymbol{\Omega}}]^{-1}\boldsymbol{\gamma} \\ \text{cov}(\boldsymbol{\xi}_*) &= K(\mathbf{t}_*, \mathbf{t}_*) \\ &\quad - K(\mathbf{t}_*, \mathbf{t})[K(\mathbf{t}, \mathbf{t}) + \Sigma_{\boldsymbol{\Omega}}]^{-1}K(\mathbf{t}, \mathbf{t}_*) \end{aligned}$$

with $\mathbf{t} = \begin{bmatrix} t_1 & t_2 & \ldots & t_n \end{bmatrix}^\top$,

$$K(\mathbf{t}_*, \mathbf{t}) = \begin{bmatrix} k(t_1, t_1) & k(t_1, t_2) & \ldots & k(t_1, t_n) \end{bmatrix}, \tag{17}$$

$$K(\mathbf{t}, \mathbf{t}_*) = K(\mathbf{t}_*, \mathbf{t})^\top \tag{18}$$

and $K(\mathbf{t}, \mathbf{t})$ is the covariance between input points.

Suppose we want to use this model for inferring velocities and acceleration. GPs are adept at predicting not only the posterior mean and covariances of the function values but their derivatives as well (Särkkä, 2011). This is because differentiation is a linear operator on the space of functions and hence the derivative of a GP is another GP. So velocity and acceleration functions obtained from applying linear operators to the position function are GPs as well. We choose to use the Square Exponential (SE) kernel given that it is analytically infinitely differentiable.

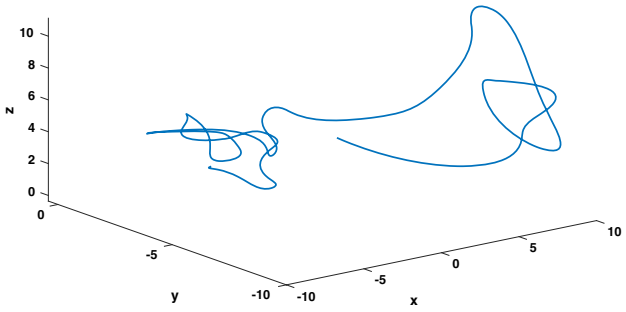


Fig. 2 3D position trajectories from the GP interpolation used to generate continuous trajectories

Consider the linear operator $\mathcal{L}^t$ applied on the function $\boldsymbol{\xi}(\mathbf{t})$ as follows,

$$\boldsymbol{\phi}(\mathbf{t}) = \mathcal{L}^t_{\phi}\boldsymbol{\xi}(\mathbf{t}) \tag{19}$$

$$\boldsymbol{\zeta}(\mathbf{t}) = \mathcal{L}^t_{\zeta}\boldsymbol{\phi}(\boldsymbol{t}) = \mathcal{L}^t_{\zeta}\mathcal{L}^t_{\phi}\boldsymbol{\xi}(\mathbf{t}), \tag{20}$$

where $\mathcal{L}^t = \mathbf{d}(\mathbf{t})$ is the derivative operator.

Note that the linear operators $\mathcal{L}^t$ are not matrix multiplication, but can be thought of as operators acting on a function and return another function with the same input domain as the input function $\phi : \mathbb{R} \mapsto \mathbb{R}$. When the operator is applied twice on the kernel (e.g., $\mathcal{L}^t_{\zeta}\mathcal{L}^t_{\phi}\boldsymbol{\xi}(\mathbf{t})$ in (20)), it is analogous of taking the partial derivative of $\boldsymbol{\xi}(\mathbf{t})$ with respect to $\mathbf{t}$ twice. Consequently, $\boldsymbol{\phi}(\mathbf{t})$ and $\boldsymbol{\zeta}(\mathbf{t})$ are GPs of the first and second derivative functions respectively. Note that when the linear operators are applied on the right-hand side of the kernel function (to propagate uncertainties) like so $\boldsymbol{\mathcal{L}}^{t_*}_{\phi}K(\mathbf{t}_*, \mathbf{t})\boldsymbol{\mathcal{L}}^t_{\phi}$, the right-hand side operator is operating on the second argument of the kernel function.

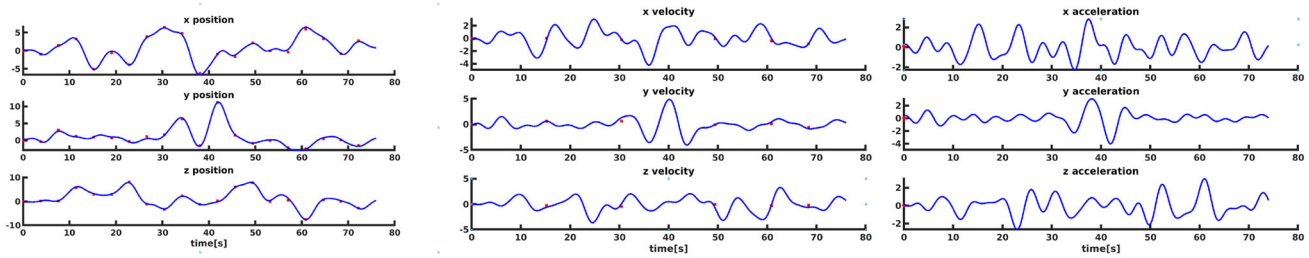
Given a set of training data including waypoints $\boldsymbol{\gamma}$, velocity $\dot{\boldsymbol{\gamma}}$, and acceleration $\ddot{\boldsymbol{\gamma}}$, a linear functional can be applied to the kernel matrix to incorporate derivative observations as,

$$\boldsymbol{\gamma}_i = \xi(\mathbf{t}) + \Omega i \tag{21}$$

$$\dot{\boldsymbol{\gamma}}_i = \boldsymbol{\mathcal{H}}^t_{\gamma}\xi(\mathbf{t}) + \Theta_i \tag{22}$$

$$\ddot{\boldsymbol{\gamma}}_i = \boldsymbol{\mathcal{H}}^t_{\gamma}\boldsymbol{\mathcal{H}}^t_{\gamma}\xi(\mathbf{t}) + \Xi_i, \tag{23}$$

where $\mathcal{H}$ is deterministic linear functional and Ω_i, Θ_i and Ξ_i are noises associated with the constraints in the position, velocity and acceleration spaces. Linear functionals are similar to linear operators, but they output vectors or matrices instead of functions.



(a) Positions over time (b) Velocities over time (c) Accelerations over time

Fig. 3 Typical position, velocity and acceleration trajectories in the (x, y, z) axes from the GP interpolation used to generate continuous trajectories, given a discrete set of constraints (red dots)

Through the application of a combination of linear operators and functionals to the kernel function of the underlying position GP function, we can conduct inference in the velocity and acceleration space (see Fig. 3). Constraints in the velocity and acceleration space, which enforce continuity at the start and end of each segment in these spaces can also be included in the measurement (waypoints) vector and multiplied properly with the kernel function through linear functionals.

The inference of $\bar{\boldsymbol{\xi}}_*$, $\bar{\boldsymbol{\phi}}_*$ and $\bar{\boldsymbol{\zeta}}_*$ with measurements in the position, velocity and acceleration spaces is given by,

$$\begin{bmatrix} \bar{\boldsymbol{\xi}}_* \\ \bar{\boldsymbol{\phi}}_* \\ \bar{\boldsymbol{\zeta}}_* \end{bmatrix} = \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma} \\ \dot{\boldsymbol{\gamma}} \\ \ddot{\boldsymbol{\gamma}} \end{bmatrix}. \tag{24}$$

Each of the terms $m_{i,j}$ in the matrix above are defined by applying the linear operator on the GP kernel in the position space, to infer both linear and angular positions, velocities and accelerations.

Position inference;

$$\begin{aligned} m_{1,1} &= K(\mathbf{t}_*, \mathbf{t})[K(\mathbf{t}, \mathbf{t}) + \Sigma_{\boldsymbol{\Omega}}]^{-1} \\ m_{1,2} &= K(\mathbf{t}_*, \mathbf{t})\mathcal{H}_\gamma^t[\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t})\mathcal{H}_\gamma^t + \Sigma_{\boldsymbol{\Theta}}]^{-1} \\ m_{1,3} &= K(\mathbf{t}_*, \mathbf{t})\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t[\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t})\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t \\ &\quad + \Sigma_{\boldsymbol{\Xi}}]^{-1}, \end{aligned}$$

Velocity inference;

$$\begin{aligned} m_{2,1} &= \mathcal{L}_\phi^{t_*} K(\mathbf{t}_*, \mathbf{t})[K(\mathbf{t}, \mathbf{t}) + \Sigma_{\boldsymbol{\Omega}}]^{-1} \\ m_{2,2} &= \mathcal{L}_\phi^{t_*} K(\mathbf{t}_*, \mathbf{t})\mathcal{H}_\gamma^t[\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t})\mathcal{H}_\gamma^t + \Sigma_{\boldsymbol{\Theta}}]^{-1} \\ m_{2,3} &= \mathcal{L}_\phi^{t_*} K(\mathbf{t}_*, \mathbf{t})\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t[\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t})\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t \\ &\quad + \Sigma_{\boldsymbol{\Xi}}]^{-1}, \end{aligned}$$

Acceleration inference;

$$\begin{aligned} m_{3,1} &= \mathcal{L}_\zeta^{t_*}\mathcal{L}_\phi^{t_*} K(\mathbf{t}_*, \mathbf{t})[K(\mathbf{t}, \mathbf{t}) + \Sigma_{\boldsymbol{\Omega}}]^{-1} \\ m_{3,2} &= \mathcal{L}_\zeta^{t_*}\mathcal{L}_\phi^{t_*} K(\mathbf{t}_*, \mathbf{t})\mathcal{H}_\gamma^t[\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t})\mathcal{H}_\gamma^t + \Sigma_{\boldsymbol{\Theta}}]^{-1} \\ m_{3,3} &= \mathcal{L}_\zeta^{t_*}\mathcal{L}_\phi^{t*} K(\mathbf{t}_*, \mathbf{t})\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t[\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t})\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t \\ &\quad + \Sigma_{\boldsymbol{\Xi}}]^{-1}. \end{aligned}$$

The covariances are given by;

$$\begin{aligned} \text{cov}(\boldsymbol{\xi}_*) &= K(\mathbf{t}_*, \mathbf{t}_*) - K(\mathbf{t}_*, \mathbf{t})[K(\mathbf{t}, \mathbf{t}) \\ &\quad + \Sigma_{\boldsymbol{\Omega}}]^{-1} K(\mathbf{t}, \mathbf{t}_*) \\ \text{cov}(\boldsymbol{\phi}_*) &= \mathcal{L}_\phi^{t_*} K(\mathbf{t}_*, \mathbf{t}_*)\mathcal{L}_\phi^{t_*} \\ &\quad - \mathcal{L}_\phi^{t_*} K(\mathbf{t}_*, \mathbf{t})\mathcal{H}_\gamma^t[\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t})\mathcal{H}_\gamma^t \\ &\quad + \Sigma_{\boldsymbol{\Theta}}]^{-1}\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t}_*)\mathcal{L}_\phi^{t_*} \\ \text{cov}(\boldsymbol{\zeta}_*) &= \mathcal{L}_\zeta^{t_*}\mathcal{L}_\phi^{t_*} K(\mathbf{t}_*, \mathbf{t}_*)\mathcal{L}_\zeta^{t_*}\mathcal{L}_\phi^{t_*} \\ &\quad - \mathcal{L}_\zeta^{t_*}\mathcal{L}_\phi^{t_*} K(\mathbf{t}_*, \mathbf{t})\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t[\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t})\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t \\ &\quad + \Sigma_{\boldsymbol{\Xi}}]^{-1}\mathcal{H}_\gamma^t\mathcal{H}_\gamma^t K(\mathbf{t}, \mathbf{t}_*)\mathcal{L}_\zeta^{t_*}\mathcal{L}_\phi^{t_*}. \end{aligned} \tag{25}$$

Note that the substitution $\mathbf{t}_* = \mathbf{t}$ is done for all instances of $\mathbf{t}_*$ after all the operations have been performed. In the equations above, $\mathbf{t}_*$ is used to remove ambiguity on which variable the operator is applied on.

5 Adaptive trace method

Following (11), the utility function I[·] evaluates the information content in the new sensor measurements with respect to the localization uncertainty. Numerous criteria exist for determining optimality in experimental designs. The most common criteria in robotics are the A-optimality and D-optimality and the choice of which one to use is made based on the application. D-optimality results in a confidence region for the parameters within the minimum volume and A-optimality minimizes the average variance or the expected mean square error (Srivastava and Anderson, 1974). For our purpose of minimizing localization uncertainty, A-optimality is the most efficient computationally, although an approximation. For any $n \times n$ matrix $\mathbf{P}$, this criterion is given by the trace of the matrix tr($\mathbf{P}$).

The utility function I[·], which is based on our proposed Adaptive trace method is then given by:

$$\mathrm{I}[\mathrm{M}(\boldsymbol{\pi}_{k:k+1})] = \begin{cases} \mathrm{tr}(\mathbf{P}_{\mathbf{b}_{k+1}}) - \mathrm{tr}(\mathbf{P}_{\mathbf{b}_k}) & \text{if} \quad \delta \geq \lambda, \\ \mathrm{tr}(\mathbf{P}_{\mathbf{r}_{k+1}}) - \mathrm{tr}(\mathbf{P}_{\mathbf{r}_k}) & \text{otherwise} \end{cases} \tag{26}$$

where $\mathbf{P_b} = [\mathbf{P}_{\mathbf{b}_f}, \mathbf{P}_{\mathbf{b}_\omega}]$ is the biases covariance matrix, $\mathbf{P_r}$ is IMU position covariance matrix, δ is the rate of change of bias uncertainty, λ is a preset threshold to determine bias uncertainty convergence, and $\pi_{k:k+1}$ is the trajectory from which evaluation measurements are obtained. Note that the algorithm keeps track of δ, even after the first convergence of the IMU bias uncertainties. Should the IMU biases uncertainties start to diverge at any stage, due to some drastic changes in the system ($\delta > \lambda$), the algorithm will switch back to the utility function that minimizes $\mathbf{P_b}$ in order to prioritize bias uncertainties convergence. This is shown in Algorithm. 1.

Algorithm 1: Switching the utility function.

```
P_b ← ∅ ;
P_r ← ∅ ;
for i = 1, ... n do
    if δ > λ then
        I[M(π_{k:k+1})] = tr(P_{b_{k+1}}) − tr(P_{b_k})
    else
        I[M(π_{k:k+1})] = tr(P_{r_{k+1}}) − tr(P_{r_k})
    end
end
return I[M(π_{k:k+1})];
```

6 Path planning

To find the trajectory that minimizes localization uncertainty in the estimated state, we define the cost function between two points as the trace of the covariance of either the biases $\mathbf{b}_f$ and $\mathbf{b}_w$, or the IMU position $\mathbf{r}$. The planner aims to excite the system in such a way that it prioritizes convergence of bias errors first then proceeds to focus on the trace of the IMU position. In order to achieve this, the planner alternates between two cases of minimizing either the IMU biases uncertainty or IMU position uncertainty based on whether or not the bias uncertainties have converged, as defined in (26). For the planner, the `Cost` function, $c(\boldsymbol{\pi}_{k:k+1})$ associated with connecting two points with a trajectory $\boldsymbol{\pi}_{k:k+1}$ can formally be defined as:

$$c(\boldsymbol{\pi}_{k:k+1}) = \mathrm{I}[\mathrm{M}(\boldsymbol{\pi}_{k:k+1})]\,, \tag{27}$$

Rapidly-exploring random tree (RRT)* (Karaman and Frazzoli, 2010) is the sampling-based motion planning algorithm used to generate a set of nodes that are used for evaluation in our framework. The nodes represent the IMU positions and orientations. The algorithm incrementally builds a tree of feasible trajectories from an initial node x_{init}. At each new iteration, a new point x_{sample} is sampled from the obstacle-free space $\mathcal{X}_{free}$, and connection attempts are made to vertices in $\mathcal{X}_{near}$, which is defined as vertices within a radius from x_{sample}. An edge is created from x_{sample} to the vertex in $\mathcal{X}_{near}$ that can be connected at a minimal cost.

The additive cost function used for evaluating nodes is defined as:

$$\begin{aligned} \texttt{Cost}(x_{\text{sample}}) = \texttt{Cost}(\text{Parent}(x_{\text{sample}})) \\ + c(\texttt{Connect}(x_{\text{sample}}, x_{\text{near}})). \end{aligned} \tag{28}$$

After the addition of the new node x_{sample} to the graph, the planner removes redundant edges from E, i.e, edges that are not part of a shortest path from x_{init}. This technique is called rewiring, and it ensures that all vertices on the tree are on a minimal cost branch. Because of the constraints the IMU modeling imposes on the system, the `Connect` function between two nodes is not a straight line as in the original RRT* algorithm. As explained above, we require smooth and continuous trajectories that are differentiable at least twice, to ensure smooth and continuous linear position, velocity and acceleration trajectories, and angular positions and velocity trajectories. We use the tailor-made interpolation method based on GP regression described in Sect. 4, that guarantees that all linear and angular position, linear and angular velocity and linear acceleration trajectories from the planner meet the continuity and smoothness constraints.

Note that 6D sampling is carried out in position and orientation as this allows us to plan in both the Cartesian and orientation spaces, and all higher order derivatives are constrained to zero. We enforce continuity by matching the linear position, velocity, acceleration and angular position and velocity at the end of a trajectory segment with those at the start of a subsequent trajectory segment in our GP regression interpolation. Additionally, the trace of the covariance matrix is used as the cost function to determine the optimal trajectory which the planner returns. This trajectory is not the shortest path but rather a trajectory which leads to quicker biases convergence with better bias estimates and ultimately better accuracy in the robot localization.

For each new node sampled by our RRT-based planner, the posterior covariance matrix $\mathbf{P}_k^+$ is initially obtained by the chosen estimation framework and propagated into the future by the linearized model using the standard equations of the Extended Kalman filter and forecasted measurements. The trace of $\mathbf{P}_k^+$ is then used for decision making by our planner. The simulated inertial measurements are used for propagation in the prediction step, while the simulated measurements from the exteroceptive sensor are taken into account during the update step. The prediction step of the filter estimates the

a-priori covariance $\mathbf{P}_k^-$, from the a-posteriori covariance estimate from the previous time step $\mathbf{P}_{k-1}^+$ where the jacobians at the current linearization point are computed from (1) and (2). Unlike traditional RRT*, we construct the tree to explore the free space and focus on information gathering, with no set goal configuration.

7 Experimental results

We validate our approach using both simulated demonstrations (Sects. 7.1, 7.2, 7.3 and 7.4) and hardware demonstrations (Sect. 8). The state estimation framework used in the simulation experiments is based on an Error State Kalman Filter (ESKF) (Sola, 2017) while ROVIO (Bloesch et al., 2015), which is an Iterated Extended Kalman Filter framework is used for the real-world experiments. The simulated acceleration, angular velocities and range measurements have simulated but realistic sensor noises added to them, $0.0196\,\mathrm{m.s}^{-2}$ and $0.0017\,\mathrm{rad.s}^{-1}$ for the accelerometer and gyroscope and 0.02 m for the range measurements.

First, we compare the resulting localization error when estimation is carried out with two different IMU bias error settings of 1% and 5% in Sect. 7.1. We then evaluate performance of our proposed adaptive trace cost function against a robot position cost function and a joint-optimization cost function in Sect. 7.2. Section 7.3, shows a comparison of the resulting bias errors when a greedy planner is used vs our proposed variant of the RRT*, with both methods using the adaptive trace cost function. Finally, in Sect. 7.4 we compare localization and bias error for the GP regression trajectories vs minimum snap trajectories, Bezier curves and B-splines trajectories over the same waypoints from our custom RRT* planner.

7.1 Evaluation of the impact of bias error on localization accuracy

In this experiment, we investigate the impact of bias error on localization accuracy. We conduct 50-run Monte-Carlo simulation with 1% and 5% bias errors to see the impact that IMU bias error has on the localization accuracy. The resulting localization error from each of these two experimental setups is shown in Fig. 4. The blue solid line represents the runs' average for the experiments with 1% IMU bias error and the red dotted line represents the runs' average for the 5% IMU bias error experiments. The shaded areas are the respective 2σ bounds.

The localization error plots show that having a higher IMU bias error in estimation negatively impacts localization accuracy. This can be seen from the considerably higher localization error of 4.822 m after 660 s in the case with 5% bias error as opposed to the 3.162 m localization error accumulated over the same time period for the 1% bias error experiment. The reason for this is the tight coupling between localization estimates and IMU bias estimates shown in (7). Prioritizing the convergence of the IMU bias estimates minimizes pose error because less biases error integrates over time.

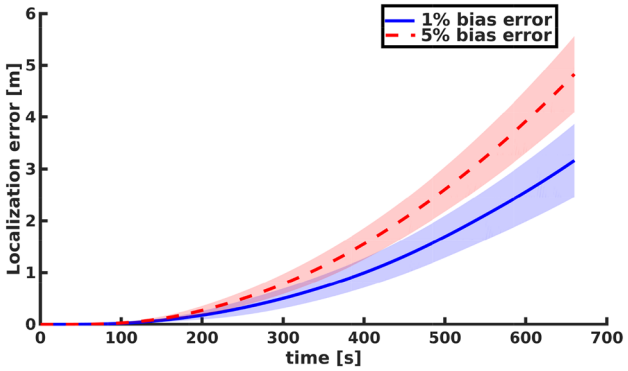


Fig. 4 Impact of 1% and 5% IMU bias error on localization accuracy. The localization errors are averaged over 50-run Monte-Carlo simulation

7.2 Evaluation of proposed adaptive trace method

In this experiment, we compare the performance of the proposed adaptive trace cost function against 2 other cost functions, one based on the position trace and the second based on joint optimization trace. The adaptive trace approach uses the trace of the bias estimate covariance until the bias uncertainties have converged. Beyond this point, the trace of the robot position estimate covariance is used for planning. The joint optimization cost function is based on work in Elisha and Indelman (2017), in which the uncertainty cost term presents a trade-off between two planning objectives of reaching a target with minimum robot position errors and calibrated sensors. The position trace cost function uses the trace of the robot position estimate covariance for planning. The results we get in terms of localization error and bias error, when these 3 cost functions are compared are shown in Figs. 5 and 6 respectively.

Initially, the localization errors of the three approaches are comparable, as can be seen in the first 200 s from the log scale plot on Fig. 5. However, with more time steps, the localization error of the adaptive traces method is less than that of the method using robot position traces alone and the joint optimization traces. This is because the adaptive traces method prioritizes convergence of the bias estimates, and better quality bias estimates ultimately lead to improved estimation of the entire state.

In both Figs. 5 and 6, the point where our method switches from using bias uncertainty to position uncertainty for planning is shown by the black arrows. After the 650 s, the average

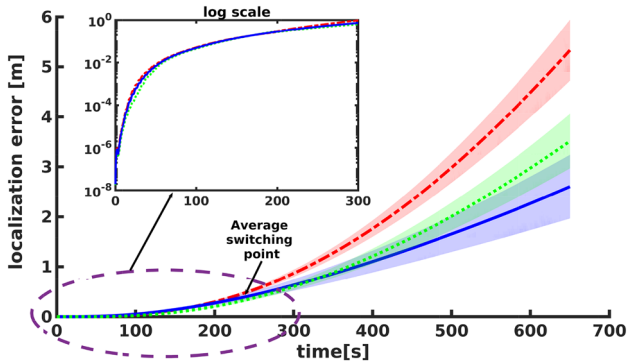


Fig. 5 Comparison of the localization errors between our proposed adaptive trace method vs a position trace method and a joint optimization trace method over a 50-run Monte-Carlo simulation

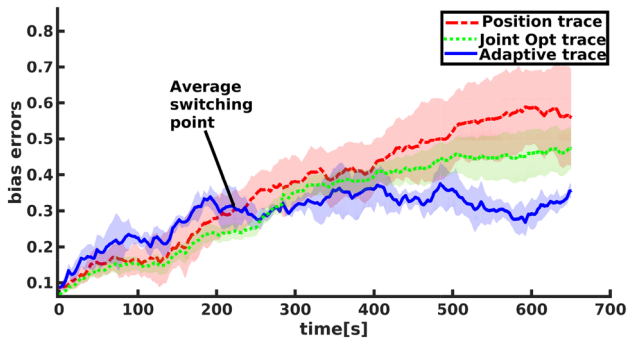


Fig. 6 Comparison of the bias errors between our proposed adaptive trace method vs a position trace method and a joint optimization trace method over a 50-run Monte-Carlo simulation. The average switching point from using biases uncertainty to position uncertainty is shown by the black arrow

localization error is 2.602 m using the adaptive trace method, 3.5 m using the joint optimization trace and 5.334 m with the robot position trace for a total travelled distance 2.6 km.

We also note how the bias convergence occurs more quickly in the approach where we inform the planner with waypoints that prioritize bias estimate. The simulation experiment under controlled conditions shows clearly the significance of the IMU bias estimation. It is clear that focusing on biases estimation first helps reducing the localization errors in the future and justifies the need to optimize the biases first. Note that for the biases error plots, the units are not included as they are a heterogeneous quantity including both the accelerometer and gyroscope bias errors which have units of ms^{-2} and $rads^{-1}$ respectively.

7.3 RRT* optimal path vs greedy planning

We compare the performance of our RRT* variant vs the greedy algorithm which only picks the best waypoint locally. Both methods use the adaptive trace as the cost function. The RRT* algorithm is limited to 3000 nodes and it is grown without a set goal node as this allows it to be more exploratory.

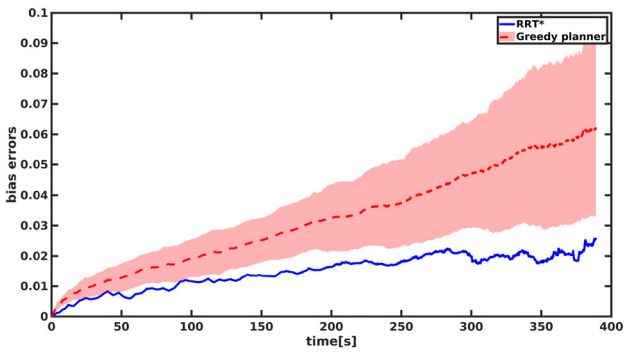


Fig. 7 Comparing of how bias errors vary over time for the greedy planner and RRT*. Both planners use the adaptive trace technique and they are run for 390 s

The biases error from the optimal RRT* path are then compared with the average bias errors from the greedy algorithm.

The results from these simulations are shown in Fig. 7. The plot shows that the bias errors from the RRT* are lower than those for the greedy planner although both planners are using the adaptive trace cost function. At the end of the 390 s trajectory, the bias error for the greedy planner is 0.062 while that for the RRT* planner is 0.026. This result is consistent with what is expected because the RRT* algorithm has a rewiring technique that ensures that the newest sample is connected on a minimal cost branch to the start node. This ultimately ensures that the RRT* algorithm finds the asymptotically optimal solution. This is different from the greedy algorithm which picks the locally optimal choice at each stage, making commitments to certain choices too early, and preventing it from finding the best overall solution later.

7.4 Comparison of trajectories from different interpolation methods

We compare the bias and localization error for acceleration trajectories from the following interpolation methods; GP regression, minimum snap, Bezier curves and B-splines, to find out which trajectories have more excitation that ultimately lead to quicker IMU bias uncertainties convergence. All methods are used to interpolate over the same waypoints generated by our custom RRT* algorithm using the adaptive trace cost function. In all four experiments, 50 Monte-Carlo runs are considered over 300 s trajectories and the average errors are compared.

The results in Table 1 show that GP interpolation performs better than minimum snap, bezier and b-spline acceleration trajectories. We believe that this is the case because the acceleration trajectories from GP regression have larger magnitudes in comparison to those from minimum snap for the same maximum acceleration setting of 2 m/s^2, as can be seen in Fig. 8 and this generates more excitation for the GP trajec-

Table 1 Comparison of the bias and robot position errors accumulated after a 300 s trajectory

Interpolation method	Average bias RMSE (m/s^2)	Average localization RMSE (m)
GP regression	**0.033**	**2.623**
minimum snap	0.0468	5.3913
Bezier curves	0.127	9.146
B-splines	0.095	7.924

The rows show the results averaged over 50 Monte-Carlo runs for the four different interpolation methods considered; Gaussian Process regression interpolation, minimum snap, Bezier curves and B-splines respectively

tories which leads to quicker convergence of the IMU bias. This ultimately leads to smaller localization errors for GP regression trajectories.

8 Hardware experiment

In the hardware experiment, we compare the performance of our proposed adaptive trace algorithm using GP regression on a UR5 arm with a stereo camera and an IMU attached as shown in Fig. 9. The aim of this experiment is to show the performance in terms of localization error, of the proposed adaptive trace method when compared to our implementation of joint optimization method used in Elisha and Indelman (2017).

The camera used in this experiment is the Realsense D455 with its internal Bosch BMI055 IMU. The camera provides global shutter RGB images at 20 Hz and IMU measurements at 200 Hz. The camera and the IMU are calibrated using Kalibr (Furgale et al., 2013). The state, feature map and the associated covariance matrix are estimated by a formulation of the Iterated Extended Kalman Filter implemented in ROVIO (Bloesch et al., 2015) after execution of our trajectories on the arm.

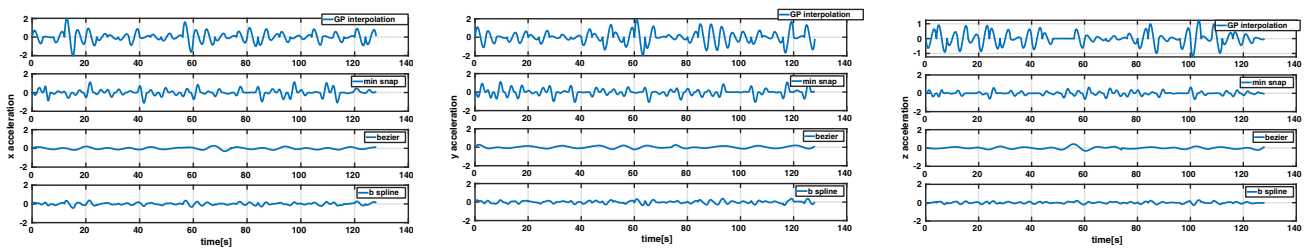
Evaluation of the information content in the trajectories generated by our planner is carried out in simulation where we simulate measurements for each of the candidate trajectories, which are evaluated by using the map of the environment and the robot state. The simulated measurements are used to propagate the filter state and its covariance. The planner then decides on the best path to execute by evaluating and comparing the information content in each of the candidate trajectories.

8.1 Robot arm planner

The planning method proposed in Sect. 6 considers trajectory samples in $SE(3)$. However, UR5 arm in our experiments has constraints on what trajectories it can execute based on its kinematic structure. For a sampled trajectory we require that a valid inverse kinematics (IK) solution exists for each pose, the joint limits of the robot are not violated and the robot does not collide with itself or the environment. Furthermore, we want to avoid any large changes in the arm's configuration between two consecutive poses in order to ensure smooth trajectories which improves camera tracking and avoids damaging the camera or its cable routed along the robot.

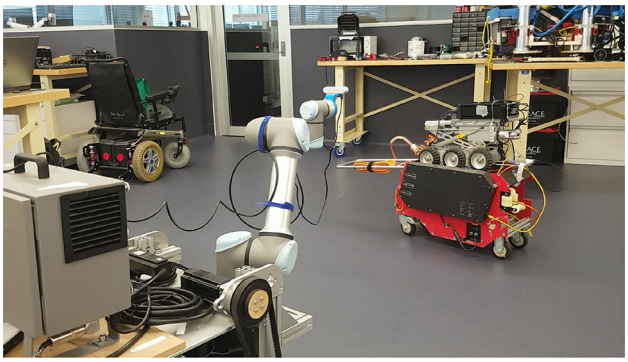
To achieve this, we leverage Hausdorff approximation planner (HAP) (Sukkar et al., 2022) which, given a robot, task-space and environment model, computes a subspace in $SE(3)$ to sample from such that the resulting executed robot trajectory satisfies our desired constraints. This subspace is represented using a discrete roadmap of poses, shown in Fig. 10, such that moving along a path between any two poses in $SE(3)$ within the subspace results in a similar length path in configuration space.

This roadmap is provided to the RRT* planner to bias its sampling towards. The sampled trajectories are then post processed and joint trajectories are generated by HAP for execution.

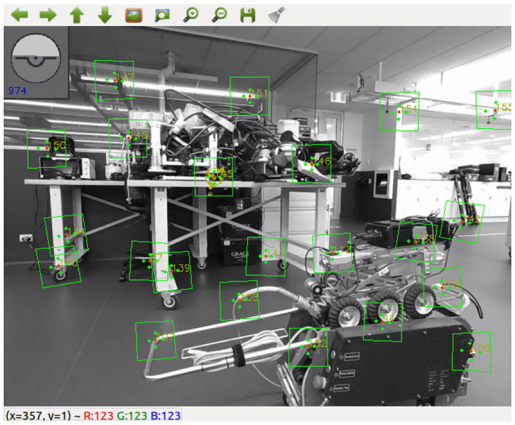


(a) Acceleration on x axis (b) Acceleration on y axis (c) Acceleration on z axis

Fig. 8 Measured accelerations on x, y and z axis from our active planning framework. The subplots show acceleration trajectories from GP interpolation, minimum snap, Bezier curves and B-splines respectively



(a)



(b)

Fig. 9 A 6-DOF UR5 arm executing an informative trajectory from our planner that prioritizes IMU bias convergence for increased localization accuracy. The arm has a Realsense D455 stereo camera with its internal Bosch BMI055 IMU attached to it, and is shown in (**a**). The visual features used by the estimation algorithm, ROVIO, are shown in (**b**)

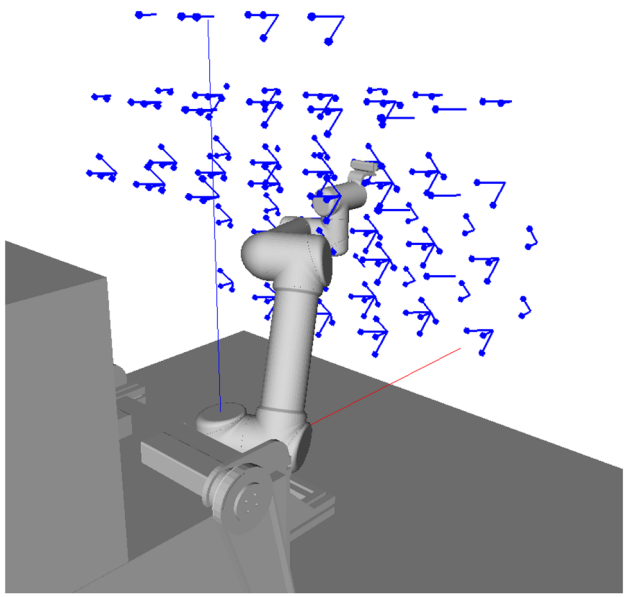


Fig. 10 The discrete roadmap of poses used by the planner to bias trajectories towards smooth paths

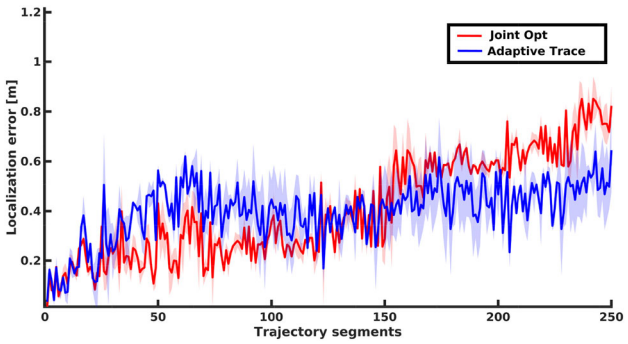


Fig. 11 Comparison of how localization error varies with the number of segments added to the trajectory for our proposed adaptive trace method vs the joint optimization method used in Elisha and Indelman (2017). An RRT* planner is used for both methods and the localization error is computed between the aligned trajectory and the arm odometry which is used as the ground truth

8.2 Adaptive trace vs joint optimization localization errors

We conducted a series of hardware experiments on the UR5 arm comparing the localization error using our proposed adaptive trace method vs the joint optimization method, for both the RRT* and the greedy planning algorithms. For these experiments, we compute the actual localization error by comparing the aligned estimated trajectory from ROVIO and the arm odometry which is used as the ground truth.

8.2.1 RRT* planner experiments

For the first set of experiments, the RRT* algorithm is used as the planner. One experiment uses the adaptive trace as the cost function for growing the tree whilst the joint optimization method is considered as the cost function in the second experiment. The adaptive approach uses the trace of the bias estimate covariance until the bias uncertainties have converged. Beyond this point, the trace of the robot position estimate covariance is used for planning. In the joint optimization method, the planner uses the trace of the whole state to pick the waypoint with the smallest trace. A sampling rate of 20 Hz is used for sampling the GP regression trajectory. For all the experiments, an identical prior trajectory is used to explore the environment first to generate a map of the environment and get a prior estimate for the state covariance. The experiments are run for the same number of trajectory segments, 250. We conduct a 5-run Monte Carlo simulation and the results of this experiment are shown in Fig. 11. The blue solid line represents the runs' average for the adaptive trace method, the red dotted line represents the runs' average for the joint optimization trace method. The shaded areas represents the respective 2σ bounds.

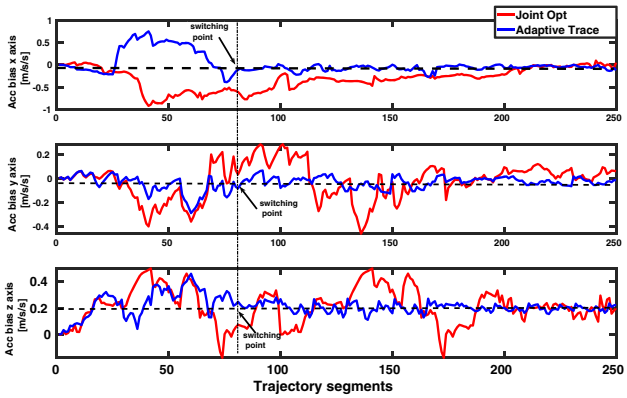


Fig. 12 Accelerometer bias error plots for the x, y and z axis. The switching point, where our adaptive trace method switches from using bias uncertainty to localization uncertainty to inform the planning is shown by the arrow. Note how the the biases converge more quickly with our adaptive trace method shown in blue compared to the joint optimization method in red

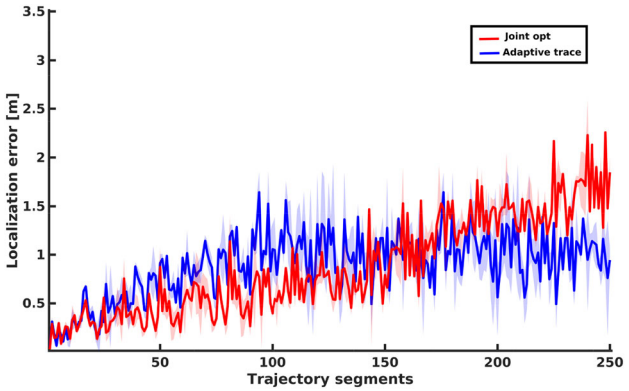


Fig. 13 Comparison of localization errors of the joint optimization method vs our proposed adaptive trace using greedy planner for both methods

The results of the experiment are shown in Figs. 11 and 12. Figure 11 shows a comparison of localization error for our adaptive trace method vs joint optimization method used in Elisha and Indelman (2017). From segment 0 to 20, the localization errors of the two methods are comparable. The joint optimization method overtakes the adaptive trace method in terms of localization beyond the 20 trajectory segment mark. However, after the accelerometer biases converge at 86 trajectory segments, the rate of increase of the localization error for the adaptive trace reduces. After the bias estimates have converged, the localization errors of the two methods are similar between 120 to 140 trajectory segments. Beyond 140 trajectory segments, our adaptive trace method outperforms the joint optimization method. The average final localization errors are $0.6451m$ and $0.832m$ for the adaptive trace and joint optimization methods respectively. This shows the accelerometer biases converge more quickly using the adaptive trace method (after 86 trajectory segments) whereas the x and z axis bias convergence occurs after 200 trajectory segments for the joint optimization method, and the y axis bias still does not converge by the end of the 250 segments considered. The accelerometer bias convergence values are shown by the black dashed line in the plots in Fig. 12 and are $[-0.1, -0.05, 0.19]$ ms^{-2} for the x y, z axis respectively. The video here youtu.be/ccGLB95opO8 shows the 3D executed informative trajectories that prioritize IMU bias convergence for increased localized accuracy as well as summarize the localization errors plot.

8.2.2 Greedy planner experiments

Similar experiments are conducted using the greedy planner. The localization errors plots for the adaptive and the joint optimization plots are similar to those in the RRT* plots. However, the accelerometer biases converge after 122 trajectory segments for the adaptive trace method and after 230 trajectory segments using a joint optimization method. The final localization errors are 0.9448 m and 1.6481 m for the adaptive trace and joint optimization methods respectively (see Fig. 13). Generally, the experiments that use RRT* planning algorithm perform better than the greedy planning algorithm (for both the adaptive trace and the joint optimization methods). This is reflected by the average localization error values at the end of the 250 trajectories considered, which are tabulated in Table 2 for the adaptive trace and joint optimization methods respectively. For both RRT* and the greedy algorithm, the biases converge quicker in the adaptive trace method than the joint optimization method. And once bias convergence occurs, localization errors for the adaptive trace grow at a slower rate than the joint optimization errors. This justifies why we optimize biases estimates first then position in our method and not the full state.

Note that for these experiments, only accelerometer biases are considered because it was found that the gyroscope biases were starting off already converged. These experiments demonstrate that planning for IMU bias convergence helps minimize localization error in state estimation. Even though our method requires a longer time horizon for it to give better performance in terms of localization error, than the joint optimization method, it ensures quicker convergence of IMU bias estimates, which ultimately gives better localization accuracy in the long run.

8.3 Real-world transferability

To show that the GP prediction corresponds to the actual executed trajectory on the arm, we calculate RMSE between the GP prediction and the executed trajectory at the end of the two trajectories. The average RMSE over 10 Monte Carlo runs was found to be 0.0246 ± 0.0032 m, with a duration of

Table 2 Comparison of the localization error using our proposed adaptive trace method vs the joint optimization method, for both the RRT* and the greedy planning algorithms

Planner and optimization method	Average localization RMSE (m)
RRT* adaptive	**0.6451**
RRT* joint optimization	0.8232
greedy adaptive	0.9448
greedy joint optimization	1.6481

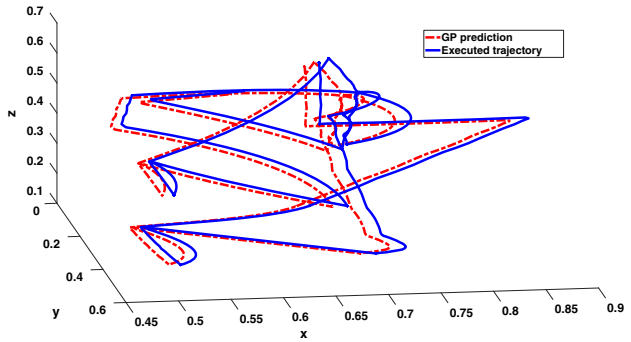


Fig. 14 GP to real world transferability

300 s for each run. The small average RMSE value after 300 s validates that the real-world transfer is valid. The RMSE value was found to be constant and did not change with the length of trajectory used. Figure 14 shows a plot GP prediction position trajectory and the trajectory of the arm followed from the arm's odometry.

9 Conclusion

This paper proposed a new algorithm for informative path planning over continuous trajectories to minimize localization error. The key contribution is the use of Gaussian Process regression to interpolate waypoints coming from our sampling based planner. Linear operators are applied to the kernel function of underlying position GP in order to infer the first and second derivative which are the velocity and acceleration respectively. The use of linear functionals enable velocity and acceleration constraints to be added in the GP model as part of the measurement vector. Although our continuous planning approach can optimize every state variable, we proposed an adaptive cost function that optimized either the robot position trace or the biases trace within the planner in order to prioritize convergence of the IMU biases.

This is because our focus was on highlighting the effect of IMU bias parameters on the localization estimation. This adaptive trace technique is used as the cost function in the RRT* variant that generates a set of discreet waypoints. Our method was evaluated in a series of simulation and real world experiments. One of the key findings of this work was that our proposed adaptive trace cost function performs better than a joint optimization trace and a robot position trace in terms of minimizing localization error and bias error. This is because the adaptive traces method prioritizes convergence of the bias estimates, and better quality bias estimates ultimately lead to improved localization.

Although there is no direct correlation between the IMU biases convergence and mapping uncertainty, mapping uncertainty is correlated to the IMU biases through the robot's pose. For future work, we plan to tackle these two problems together, to minimize mapping uncertainty whilst also reducing localization uncertainty through IMU bias convergence. Reformulating our IPP utility to satisfy these two objectives simultaneously would be solving the active SLAM problem.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10514-024-10180-6.

Author Contributions M.U. and T.V.C. conceived of the presented idea and developed the theoretical formalism. M.U. carried out the implementation and conducted the simulation and hardware experiments. F.S. assisted in setting up the hardware experiments and worked on the robot arm planner. C.Y. and R.F. provided early feedback and supervision, and T.V.C. oversaw the full project. All authors offered critical feedback and contributed to shaping the research, analysis, and manuscript.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions This research is supported by an Australian Government Research Training Program (RTP) Scholarship, the University of Technology Sydney and the Industrial Transformation Training Centre (ITTC) for Collaborative Robotics in Advanced Manufacturing (also known as the Australian Cobotics Centre) funded by ARC (Project ID: IC200100001)

Data availability No datasets were generated or analysed during the current study.

Declarations

Conflict of interest We declare that the authors have no Conflict of interest as defined by Springer, or other interests that might be perceived to influence the results and/or discussion reported in this paper.

Ethical approval Not applicable

References

Bähnemann, R., Burri, M., Galceran, E., Siegwart, R., & Nieto, J. (2017). Sampling-based motion planning for active multirotor system identification. In *2017 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 3931–3938).

Bai, N., Tian, Y., Liu, Y., Yuan, Z., Xiao, Z., & Zhou, J. (2020). A high-precision and low-cost IMU-based indoor pedestrian positioning technique. *IEEE Sensors Journal, 20*(12), 6716–6726.

Bloesch, M., Omari, S., Hutter, M., & Siegwart, R. (2015). Robust visual inertial odometry using a direct EKF-based approach. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 298–304).

Burke, D., Chapman, A., & Shames, I. (2021). Fast spline trajectory planning: Minimum snap and beyond. arXiv preprint arXiv:2105.01788.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., et al. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics, 32*(6), 1309–1332.

Chang, H., Yang, W., Zhang, H., Yang, X., & Chen, C. Y. (2016). An improved FastSLAM using resampling based on particle swarm optimization. In *2016 IEEE 11th Conference On Industrial Electronics and Applications (ICIEA)* (pp. 229–234).

Elisha, Y.B., & Indelman, V. (2017). Active online visual-inertial navigation and sensor calibration via belief space planning and factor graph based incremental smoothing. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (iros)* (pp. 2616–2622).

Furgale, P., Rehder, J., & Siegwart, R. (2013). Unified temporal and spatial calibration for multi-sensor systems. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 1280–1286).

Hausman, K., Preiss, J., Sukhatme, G. S., & Weiss, S. (2017). Observability-aware trajectory optimization for self-calibration with application to UAVS. *IEEE Robotics and Automation Letters, 2*(3), 1770–1777.

Hitz, G., Galceran, E., Garneau, M. È., Pomerleau, F., & Siegwart, R. (2017). Adaptive continuous-space informative path planning for online environmental monitoring. *Journal of Field Robotics, 34*(8), 1427–1449.

Karaman, S., & Frazzoli, E. (2010). Incremental sampling-based algorithms for optimal motion planning. *Robotics Science and Systems VI, 104*(2).

Le Gentil, C., & Vidal-Calleja, T. (2023). Continuous latent state preintegration for inertial-aided systems. *The International Journal of Robotics Research, 42*(10), 874–900.

Li, M., & Mourikis, A. I. (2014). Online temporal calibration for camera-imu systems: Theory and algorithms. *The International Journal of Robotics Research, 33*(7), 947–964.

Li, M., Yu, H., Zheng, X., & Mourikis, A. I. (2014). High-fidelity sensor modeling and self-calibration in vision-aided inertial navigation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 409–416).

Liu, B., Adams, M., & Ibañez-Guzmán, J. (2005). Minima controlled recursive averaging noise reduction for multi-aided inertial navigation of ground vehicles. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 3408–3414).

Marchant, R., & Ramos, F. (2014). Bayesian optimisation for informative continuous path planning. In *2014 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 6136–6143).

Maye, J., Sommer, H., Agamennoni, G., Siegwart, R., & Furgale, P. (2016). Online self-calibration for robotic systems. *The International Journal of Robotics Research, 35*(4), 357–380.

Mellinger, D., & Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation* (pp. 2520–2525).

Mellinger, D., Kushleyev, A., & Kumar, V. (2012). Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams. In *2012 IEEE International Conference on Robotics and Automation* (pp. 477–483).

Mukadam, M., Dong, J., Yan, X., Dellaert, F., & Boots, B. (2018). Continuous-time Gaussian process motion planning via probabilistic inference. *The International Journal of Robotics Research, 37*(11), 1319–1340.

Murali, V., Nieto, C., Choudhary, S., & Christensen, H. I. (2016). Active planning based extrinsic calibration of exteroceptive sensors in unknown environments. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 2498–2505).

Panigrahi, P. K., & Bisoy, S. K. (2021). Localization strategies for autonomous mobile robots: A review. *Journal of King Saud University-Computer and Information Sciences*.

Papachristos, C., Khattak, S., & Alexis, K. (2017). Autonomous exploration of visually-degraded environments using aerial robots. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)* (pp. 775–780).

Perera, L. D. L., Wijesoma, W. S., & Adams, M. D. (2006). The estimation theoretic sensor bias correction problem in map aided localization. *The International Journal of Robotics Research, 25*(7), 645–667.

Popović, M., Vidal-Calleja, T., Hitz, G., Chung, J. J., Sa, I., Siegwart, R., & Nieto, J. (2020). An informative path planning framework for uav-based terrain monitoring. *Autonomous Robots, 44*(6), 889–911.

Qin, H., Meng, Z., Meng, W., Chen, X., Sun, H., Lin, F., & Ang, M. H. (2019). Autonomous exploration and mapping system using heterogeneous UAVS and UGVS in GPS-denied environments. *IEEE Transactions on Vehicular Technology, 68*(2), 1339–1350.

Särkkä, S. (2011). Linear operators and stochastic partial differential equations in Gaussian process regression. In *International Conference on Artificial Neural Networks* (pp. 151–158).

Schneider, T., Li, M., Cadena, C., Nieto, J., & Siegwart, R. (2019). Observability-aware self-calibration of visual and inertial sensors for ego-motion estimation. *IEEE Sensors Journal, 19*(10), 3846–3860.

Siegwart, R., Nourbakhsh, I. R., & Scaramuzza, D. (2011). *Introduction to Autonomous Mobile Robots*. MIT Press.

Sola, J. (2017). Quaternion kinematics for the error-state Kalman filter. arXiv preprint arXiv:1711.02508

Srivastava, J., & Anderson, D. (1974). A comparison of the determinant, maximum root, and trace optimality criteria. *Communications in Statistics-Theory and Methods, 3*(10), 933–940.

Sukkar, F., Wakulicz, J., Lee, K. M. B., & Fitch, R. (2022). Motion planning in task space with Gromov-Hausdorff approximations. *International Journal of Robotics Research*.

Usayiwevu, M. (2022). Active perception for inertial-aided systems. University of Technology Sydney (Australia).

Usayiwevu, M., Le Gentil, C., Mehami, J., Yoo, C., Fitch, R., & Vidal-Calleja, T. (2020). Information driven self-calibration for Lidar-

inertial systems. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 9961–9967).

Usayiwevu, M., Sukkar, F., & Vidal-Calleja, T. (2023). Probabilistic plane extraction and modeling for active visual-inertial mapping. In *2023 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 10601–10607).

Wakulicz, J., Lee, K.M.B., Yoo, C., Vidal Calleja, T., & Fitch, R. (2022). Informative planning for worst-case error minimisation in sparse gaussian process regression. In *IEEE International Conference on Robotics and Automation*.

Webb, D. J., Crandall, K. L., & van den Berg, J. (2014). Online parameter estimation via real-time replanning of continuous gaussian pomdps. In *2014 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 5998–6005).

Williams, C. K., & Rasmussen, C. E. (2006). *Gaussian Processes for Machine Learning* (Vol. 2). Cambridge, MA: MIT Press. No. 3.

Yang, Y., Geneva, P., Eckenhoff, K., & Huang, G. (2019). Degenerate motion analysis for aided ins with online spatial and temporal sensor calibration. *IEEE Robotics and Automation Letters, 4*(2), 2070–2077.

Yang, Y., Geneva, P., Zuo, X., & Huang, G. (2020). Online imu intrinsic calibration: Is it necessary? In *2020 Robotics: Science and Systems*.

Yang, Y., Geneva, P., Zuo, X., & Huang, G. (2023). Online self-calibration for visual-inertial navigation: Models, analysis, and degeneracy. *IEEE Transactions on Robotics, 39*(5), 3479–3498.

Yassine, A., Fawzi, S., Khalil, M., et al. (2022). A robust synergetic controller for quadrotor obstacle avoidance using Bézier curve versus b-spline trajectory generation. *Intelligent Service Robotics*, pp. 1–10.