JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 39, No. 2, February 2016

Continuous-Time Estimation of Attitude Using B-Splines on Lie Groups

Hannes Sommer*
Swiss Federal Institute of Technology Zurich, 8092 Zurich, Switzerland
James Richard Forbes†
University of Michigan, Ann Arbor, Michigan 48109-2140
and
Roland Siegwart‡ and Paul Furgale§
Swiss Federal Institute of Technology Zurich, 8092 Zurich, Switzerland

DOI: 10.2514/1.G001149

Filtering algorithms are the workhorse of spacecraft attitude estimation, but recent research has shown that the use of batch estimation techniques can result in higher accuracy per unit of computational cost. This paper presents an approach for singularity-free batch estimation of attitude in continuous time using B-Spline curves on unit-length quaternions. It extends existing theory of unit-length quaternion B-splines to general Lie groups and arbitrary B-spline order. It is shown how to use these curves for continuous-time batch estimation using Gauss–Newton or Levenberg–Marquardt, including efficient curve initialization, a parameter update step that preserves the Lie group constraint within an unconstrained optimization framework, and the derivation of Jacobians of the B-spline's value and its time derivatives with respect to an update of its parameters. For unit-length quaternion splines, the equations for angular velocity and angular acceleration are derived. An implementation of this algorithm is evaluated on two problems: spacecraft attitude estimation using a three-axis magnetometer, a sun sensor, and 1) a three-axis gyroscope, and 2) a continuous-time vehicle dynamics model based on Euler's equation. Its performance is compared against a standard multiplicative extended Kalman filter and a recently published batch attitude estimation algorithm. The results show that B-splines have equal or superior performance over all test cases and provide two key tuning parameters, the number of knots and the spline order, that an engineer can use to trade off accuracy and computational efficiency when choosing a spline representation for a given estimation problem.

Nomenclature

$\mathcal{A}$	=	associative algebra and ambient space of $\mathcal{G}$
$\mathcal{A}_{\text{inv}}$	=	group of multiplicative invertible elements in $\mathcal{A}$
$\mathcal{G}$	=	general Lie group
$\mathfrak{g}$	=	$\mathcal{G}$'s Lie algebra
$\boldsymbol{g}$	=	element of $\mathcal{G}$
$\mathbb{H}$	=	skew field of quaternions
I	=	number of knots in the B-spline
$\boldsymbol{\imath}$	=	multiplicative identity element
K	=	$I - O$, number of control vertices in the B-spline
O	=	B-spline order
$\boldsymbol{q}$	=	unit-length quaternion
$\mathbb{R}$	=	field of real numbers
Rot_q	=	rotation associated with $\boldsymbol{q}$
T	=	end time
$\mathcal{T}$	=	time range $[0, T]$
$\|\cdot\|$	=	Euclidean 2-norm in $\mathbb{R}^n$, for any $n \in \mathbb{N}$

I. Introduction

RECENT work on online state estimation in robotics has moved away from filtering algorithms toward batch processing based on Gauss–Newton. Strasdat et al. [1] showed that, using modern sparse matrix methods, batch processing results in higher accuracy per unit of computational work. This move toward batch processing has resulted in a number of new approaches to state estimation such as efficient and accurate iterative fixed-lag smoothing [2] and incremental batch estimation [3]. The benefits of treating data in batch were clearly shown by Vandersteen et al. [4] who presented a moving-horizon estimator for spacecraft attitude estimation based on an efficient sparse matrix solution to the Gauss–Newton problem. Although this method shows excellent results when compared to filtering algorithms, it requires an estimate of the attitude at each measurement time, limiting it to low-rate sensors.

The requirement to estimate the state at each measurement time is fundamentally tied to the discrete-time approximation made by the vast majority of estimators. However, Furgale et al. [5] proposed an alternate approach, leaving the batch estimation problem in continuous time and approximating the state as a weighted sum of a finite number of temporal basis functions (such as a B-spline curve). A primary benefit to this approach is that it decouples the number of parameters in the estimation problem from the number of measurements as the vehicle state may be queried at any time.

In discrete time, the question of how to parameterize a three-degree-of-freedom orientation, a member of the noncommutative group SO(3), has been subject of decades of research and debate. Algorithms such as the Q-method [6], the QUEST algorithm [7], and the more recent ESOQ [8] and ESOQ2 [9], which are batch methods that are computationally light and suitable for online and real-time use, use unit-length quaternions. Although one of the first applications of the Kalman filter employed Euler angles [10], the vast majority of Kalman-type filters, i.e., the extended Kalman filter (EKF) or unscented Kalman filter (UKF), use unit-length quaternions or a combination of unit-length quaternions and Gibbs parameters [11–14]. The question is no less important in continuous time, where we

Received 4 November 2014; revision received 13 April 2015; accepted for publication 18 April 2015; published online 10 August 2015.

*Ph.D. Candidate, Autonomous Systems Lab, LEE Building, Leonhardstrasse 21; hannes.sommer@mavt.ethz.ch.
†Assistant Professor, Department of Aerospace Engineering, FXB Building, 1320 Beal Street; forbesrj@umich.edu. Member AIAA.
‡Professor, Autonomous Systems Lab, LEE Building, Leonhardstrasse 21; rsiegwart@ethz.ch.
§Senior Researcher, Autonomous Systems Lab, LEE Building, Leonhardstrasse 21; paul.furgale@mavt.ethz.ch.

must decide how to parameterize a continuously time-varying orientation. Furgale et al. [5] use the simple approach of defining a 3×1 B-spline of Cayley–Gibbs–Rodrigues parameters [15]. The B-spline provides analytical formulas for parameter rates, allowing the computation of angular velocity at any point ([16] Table 2.3).

However, analogously to the discrete-time case, any curve through the parameter space of a minimal attitude parameterization will suffer from a number of problems. In the absence of other information, a batch estimator will produce an answer that takes the shortest distance in parameter space, which is not necessarily the same as the shortest distance in the space of rotations. Consequently, the estimate produced may be dependent on the coordinate frame in which we choose to express the problem because this coordinate frame decides what part of parameter space the answer lives in. Furthermore, every minimal parameterization of rotation has a singularity, and so, when using this approach, there may be a danger of approaching this singularity during the estimation process.

Kim et al. [17] propose a method of generating B-splines on SO(3) using unit-length quaternions and their associated exponential map. The curves generated by this approach are valid unit-length quaternions at every point, and time derivatives of the curve are found by a straightforward use of the properties of the exponential map. Consequently, we would like to evaluate the approach proposed in Furgale et al. [5] with the one proposed in Kim et al. [17]. To do so, we must further develop the theory presented in Kim et al. to be suitable for estimation. Specifically, we offer the following contributions.

1) We extend the unit-length quaternion B-spline approach of Kim et al. [17] to apply to any Lie group and present all derivations needed to build an estimator for a broad subclass of Lie groups.

2) We prove that this construction has the desirable property of biequivariance (applying a group operation from the left and right to each control vertex applies this same operation to every part of the curve), which ensures that the result of the estimation is independent of the chosen coordinate system.

3) We present a method of including arbitrary nonlinear continuous-time motion models in the estimator.

4) We develop a method of initializing the spline control vertices to act as an initial guess for batch, nonlinear minimization.

5) We derive the specific equations for unit-length quaternion curves including a) the equation for angular velocity of a body, b) the equation for angular acceleration of a body, and c) analytical Jacobians that relate small changes in the (unit-length quaternion) spline control vertices to small changes in orientation, angular velocity, and angular acceleration.

6) Finally, we compare a family of spline-based estimators to standard approaches on two simulated spacecraft-attitude-estimation problems: one using measured dynamics from a gyroscope, and the other using dynamics based on Euler's equation. In both cases, we compare against a standard multiplicative extended Kalman filter (MEKF) and the batch discrete-time estimator recently proposed by Vandersteen et al. [4].

II. Related Work

An increasing body of literature in robotics shows that state estimation based on batch optimization has a higher accuracy than filtering approaches per unit of computational work. A primary study in this is provided by Strasdat et al. [1], who show that, for the particular problem of camera-based simultaneous localization and mapping, keyframe-based optimization that chooses a subset of informative measurements outperforms filtering. Similar results have been shown for other problems, especially the fusion of visual and inertial measurements. Leutenegger et al. [2] directly use nonlinear optimization for online visual inertial sensor fusion, showing higher accuracy than an EKF approach while still running at sensor rate. Mourikis et al. [18] take a different approach, proposing a filtering framework that keeps multiple clones of the state and uses batch optimization of landmark locations as an efficient update step for spacecraft entry, descent, and landing. Their follow-on work shows the clear benefit of optimizing over temporally consecutive state clones when compared to pure filtering [19].

These "online batch" algorithms follow the established formula for continuous-time filtering with discrete measurements first proposed by Moore and Tam [20]. For the continuous system dynamics, they use measured dynamics (from the inertial measurement unit) in the place of the filter's control input and implement an integration scheme between exteroceptive observations. This inherently links the size of the state vector to the number of observations. For some problems, there is no drawback to this coupling. For others, such as the alignment of pushbroom imagery from satellites (see Poli and Toutin [21] and references therein), the processing of data from rolling shutter cameras [22–24], or continuously moving sweeping laser scanners [25–28], an estimate of the state is required at such a high rate that a discrete-time formulation becomes intractable for all but the smallest problems. To maintain a tractable estimation problem, each of these papers uses a continuous-time state representation to decouple the size of the state vector from the measurement times, either parametric curves [21,22,24–27] or Gaussian process (GP) regression [28]. These two competing approaches both consider all measurements at once, and they both use Gauss–Newton to derive a maximum likelihood estimate for the state variables. The key difference is on the state representation.

Tong et al. [29] describe the Gaussian process Gauss–Newton (GPGN) algorithm, which uses a finite set of samples of the state at different times as the set of parameters to estimate. The GP prediction equation is used to look up the state in between the sample times. The GP covariance function acts as a regularizer to keep the state estimate smooth. This approach has three shortcomings. First, the GP prediction equations require one to build and invert a dense matrix the size of the state vector, limiting the applicability to large problems. Second, the covariance function of the GP, which defines the smoothness of the solutions, must be chosen ahead of time and does not represent a physical process. Hence, it is currently unknown how to incorporate nonlinear continuous-time systems models into GPGN. Finally, it is not yet clear how to define a GP and a covariance function on a Lie group such as SO(3). For these reasons, we pursue a parametric approach, which allows us to naturally handle each of these issues.

The parametric curve approach uses a weighted sum of known temporal basis functions to represent the state. Although there have been many isolated publications using parametric curves over the years, Furgale et al. [5] showed how it relates back to continuous-time maximum likelihood estimation [30], presented the mathematics in their most general basis-function form, and expanded the derivation to include an arbitrary nonlinear continuous-time system model. However, in this and follow-on work, they limited their applications to problems where it was possible to use curves in attitude parameter space and simple, white-noise motion system models (cf. Oth et al. [24] and Furgale et al. [31]).

Continuous-time representation of attitude has mostly used either curves through parameter space [5,21,24,31] or spherical linear interpolation on SO(3) [22,23,26]. Linear interpolation is singularity-free, but curves are only C^0 continuous, which is not a good fit for smooth vehicle motion. Anderson and Barfoot [27] model the trajectory as a curve through velocity space, which is able to represent attitude change with no singularities, but requires numerical integration of the curve if the pose is required. Kim et al. [17] present a general approach for cubic B-spline unit quaternion curves but their target was computer graphics. This approach was adapted for state estimation by Lovegrove et al. [32], who use the same curve construction technique to build curves on SE(3). In contrast, we extend this curve construction scheme to arbitrary B-spline order (second order corresponds to spherical linear interpolation) and provide general formulas for the time derivatives and Jacobians needed for optimization.

To the best of our knowledge, this is the first work to include an arbitrary nonlinear system model into basis-function formulated state estimation. A similar optimization process was previously proposed by [33] for trajectory generation for mobile robot planning on rough terrain. They estimated a polynomial curve through the control inputs of a wheeled mobile robot that would take the robot from the current pose to the goal pose. They forward-simulated the vehicle dynamics using numerical integration and used the error between the final pose and the goal pose to iteratively update the polynomial coefficients until convergence. In contrast, we assume the control inputs are

known and use a numerical integration scheme to apply the system model to the parameterized state.

Spacecraft attitude estimation is an ideal problem to test advances in the mathematical representation of orientation, and for this reason, it has been at the center of attitude estimation research for decades. Early spacecraft-centric batch attitude determination methods, such as the Q-method [6] and QUEST [7], are solutions to Wahba's problem [34]. The Q-method and QUEST, as well as ESOQ and ESOQ2 [8,9], employ unit-length quaternions when estimating attitude. Other solutions to Wahba's problem, such as Farrel et al. [35], Markley [36], and Forbes and de Ruiter [37], estimate the attitude rotation matrix rather than an attitude parameterization. Wahba's problem in SO(n) is considered in de Ruiter and Forbes [38]. There is a vast literature devoted to spacecraft attitude estimation using Kalman-type filters such as the EKF and the UKF [39]. Unit-length quaternions are the preferred attitude description when Kalman filtering due to the absence of singularities; for example, see Lefferts et al. [11], Bar-Itzhack and Oshman [12], Shuster [13], Crassidis and Markley [14], Bar-Itzhack and Reiner [40], and Choukroun et al. [41]. Often a three-parameter attitude representation, such as Gibbs parameters, are used together with a quaternion in a multiplicative framework [14,42,43]. Rather than trying to circumvent various issues associated the unit-length quaternion constraint within multiplicative filtering structure, Zanetti et al. [44] and Chee and Forbes [45] revisit the derivation of the discrete-time Kalman filter and directly incorporate the unit-length quaternion constraint (which is in effect a norm constraint) into the derivation. Normalization of an unconstrained estimate is shown to be optimal. Forbes et al. [46] presents a continuous-time generalization of Zanetti et al. [44]. Various authors have also considered estimating the rotation matrix directly in both stochastic and deterministic settings. Bar-Itzhack and Reiner [40] and Choukroun et al. [47,48] consider rotation matrix estimation within a Kalman filtering framework. Khosravian and Namvar [49] and Firoozi and Namvar [50] consider rotation matrix estimation of a spacecraft endowed with a rate gyro and one vector measurement, generalizing the SO(3) estimator structure presented in [51–54]. Similar rotation matrix estimators can be found in Kinsey and Whitcomb [55] and Grip et al. [56].

The closest work to ours is that of Vandersteen et al. [4], who propose a batch discrete time estimation framework for attitude estimation and spacecraft calibration. They show that batch or moving horizon estimates outperform filters in terms of accuracy and convergence rate. However, in this formulation, the number of states is tied to the number of measurement times. This limits the approach to low-rate sensors.

III. Theory

This section will present theoretical background and contributions. We will assume that the reader is familiar with the basics of B-splines and the very basics of Lie groups. For a thorough introduction to B-splines, see Bartels et al. [57]. For a brief overview with a focus on estimation, see Furgale et al. [5]. For a thorough introduction to Lie groups, see Kirillov et al. [58], and for a more applied approach, see Hall [59].

A. Continuous-Time Batch State Estimation

This section will briefly review the continuous-time batch state estimation framework used in this paper and originally presented in Furgale et al. [5]. The probabilistic derivation for estimation of a continuous-time process $\boldsymbol{x}(t)$ with discrete measurements seeks an estimate of the joint posterior density $p(\boldsymbol{x}(t)|\boldsymbol{u}(t), \boldsymbol{z}_{1:N})$ over the interval $\mathcal{T} = [0, T]$, where $\boldsymbol{u}(t)$ is a control input to the system, and $\boldsymbol{z}_i$ is a measurement taken at time $t_i \in \mathcal{T}$, $1 \leq i \leq N$. Assuming the probability density of the initial state, $p(\boldsymbol{x}(t_0))$, is known, this density is commonly factored as

$$p(\boldsymbol{x}(t)|\boldsymbol{u}(t), \boldsymbol{z}_{1:N}) = \frac{p(\boldsymbol{x}(t)|\boldsymbol{u}(t)) \prod_{i=1}^{N} p(\boldsymbol{z}_i|\boldsymbol{x}(t_i))}{p(\boldsymbol{z}_{1:N})} \tag{1}$$

by using Bayes's rule and assuming that the measurements' distributions given the state 1) do not depend on the control inputs, and 2) are mutually independent.

By assuming that the densities involved are Gaussian, we can write the measurement model:

$$\boldsymbol{z}_i = \boldsymbol{h}_i(\boldsymbol{x}(t)) + \boldsymbol{n}_i, \qquad \boldsymbol{n}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_i) \tag{2}$$

and measurement distribution:

$$p(\boldsymbol{z}_i|\boldsymbol{x}(t)) = \mathcal{N}(\boldsymbol{h}_i(\boldsymbol{x}(t)), \boldsymbol{R}_i) \tag{3}$$

where i refers to the ith measurement; $\boldsymbol{h}_i(\cdot)$ is a nonlinear measurement model, typically evaluating $\boldsymbol{x}$ and its derivatives at a measurement time t_i; and $\boldsymbol{R}_i$ is the covariance matrix of the measurement noise.

We may also write down the process model as a continuous stochastic dynamical system ([30] p. 143) described formally by the differential equation:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) + \boldsymbol{w}(t), \qquad \boldsymbol{w}(t) \sim \mathcal{GP}(\boldsymbol{0}, \boldsymbol{Q}\delta(t - t')) \tag{4}$$

where the overdot represents the time derivative; $\boldsymbol{f}(\cdot)$ is a deterministic function; $\boldsymbol{w}(t)$ is a zero-mean, white Gaussian process with power-spectral-density matrix $\boldsymbol{Q}$; and $\delta(\cdot)$ is Dirac's delta function. The notation denotes a Gaussian process in the same way that $\mathcal{N}(\cdot)$ denotes a Gaussian distribution [60]. The process distribution may then be written as ([30] p. 156)

$$p(\boldsymbol{x}(t)|\boldsymbol{u}(t)) \propto p(\boldsymbol{x}(t_0)) \exp\left(-\frac{1}{2}\int_0^T \boldsymbol{e}_u(\tau)^T \boldsymbol{Q}^{-1} \boldsymbol{e}_u(\tau)\, \mathrm{d}\tau\right) \tag{5}$$

where

$$\boldsymbol{e}_u(t) \coloneqq \dot{\boldsymbol{x}}(t) - \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t) \tag{6}$$

A batch estimator for $\boldsymbol{x}(t)$ may be derived by taking the negative logarithm of the posterior likelihood:

$$\{\boldsymbol{x}(t)^\star\} = \underset{x(t)}{\operatorname{argmin}}(-\log(p(\boldsymbol{x}(t)|\boldsymbol{u}(t), \boldsymbol{z}_{1:N})) \tag{7}$$

The key innovation presented in Furgale et al. [5] was to leave the problem in continuous time and make estimation of Eq. (7) tractable by modeling the process as a weighted sum of a finite number of known analytical basis functions:

$$\boldsymbol{\Phi}(t) \coloneqq [\boldsymbol{\phi}_1(t) \ldots \boldsymbol{\phi}_M(t)], \qquad \boldsymbol{x}(t) \coloneqq \boldsymbol{\Phi}(t)\boldsymbol{c} \tag{8}$$

where each $\boldsymbol{\phi}_m(t)$ is a $D \times 1$ basis function, $\boldsymbol{\Phi}(t)$ is a $D \times M$ stacked basis matrix, and $\boldsymbol{c}$ is a $M \times 1$ column of coefficients. Hence, by making the substitution in Eq. (8), we have turned a difficult problem (estimate a process at the infinite number of points in $\mathcal{T}$) to a simpler problem that we know how to solve: estimate the column of coefficients $\boldsymbol{c}$. We call this the basis function formulation:

$$\begin{aligned}\{\boldsymbol{c}^\star\} &= \underset{c}{\operatorname{argmin}}(-\log(p(\boldsymbol{x}(t)|\boldsymbol{u}(t), \boldsymbol{z}_{1:N})) \\ &\approx \underset{x(t)}{\operatorname{argmin}}(-\log(p(\boldsymbol{x}(t)|\boldsymbol{u}(t), \boldsymbol{z}_{1:N}))\end{aligned} \tag{9}$$

where the approximation sign indicates that we are approximating the process $\boldsymbol{x}(t)$ with Eq. (8). This problem is easily solved using standard methods like Gauss–Newton or Levenberg–Marquardt.

The basis function formulation works very well when the process lives in a vector space. However, this does not cover common cases in spacecraft attitude estimation or robotics, where vehicle states usually are of non-vector space type like the Lie groups SO(3) or SE(3). The next section derives one approach to address this issue.

B. Construction of a Lie-Group-Valued B-Spline

In this section, we derive a method to construct a B-spline curve on a finite-dimensional Lie group. Our derivation is based on the

unit-length quaternion B-spline curve from Kim et al. [17]. They define a curve based on cumulative B-spline basis functions. Cumulative basis functions represent an alternative but equivalent method of constructing a B-spline function on $\mathbb{R}$-vector spaces. The derivation presented later is valid for any nondecreasing knot sequence.

We will quickly motivate that alternative form. Let $\boldsymbol{f}$ be a B-spline function of order O defined on a nondecreasing sequence of knots, $(t_i)_{i=1}^{I}$ with $I \geq 2O$. At time $t \in [t_i, t_{i+1})$, $\boldsymbol{f}$, when $i \geq O$ and $i \leq K \coloneqq I - O$, may be written as

$$\boldsymbol{f}(t) = \sum_{j=s(i)}^{i} b_j(t)\boldsymbol{c}_j \tag{10}$$

where $s(i) \coloneqq i - (O-1)$, $(\boldsymbol{c}_j)_{j=1}^{K} \in \mathcal{V}$ denotes the control vertices in an $\mathbb{R}$-vector space $\mathcal{V}$, and $b_j \coloneqq B_{j,O}$ denotes the jth B-spline basis function, where $B_{j,O}$ is defined as in [17] or de Boor [61]. The B-spline order O is precisely the spline's polynomial degree plus 1. This construction, (10), is illustrated in Fig. 1 for the case $\dim \mathcal{V} = 1$ and $O = 4$. For $t < t_O$ and $t \geq t_{I-O+1}$ (outside the gray shaded area in Fig. 1), there are fewer than O nonzero basis functions, and we treat the B-spline as undefined and assume for the rest of the paper that $t \in [t_O, t_{I-O+1})$. This expression may be rearranged into the cumulative form as follows

$$\begin{aligned}\boldsymbol{f}(t) &= \boldsymbol{c}_{s(i)} + \sum_{j=s(i)+1}^{i}\left(\sum_{k=j}^{i} b_k(t)\right)(\boldsymbol{c}_j - \boldsymbol{c}_{j-1}) \\ &= \boldsymbol{c}_{s(i)} + \sum_{j=1}^{O-1}\beta_{i,j}(t)(\boldsymbol{c}_{s(i)+j} - \boldsymbol{c}_{s(i)+j-1})\end{aligned} \tag{11}$$

defining the cumulative basis functions

$$\beta_{i,j}(t) \coloneqq \sum_{k=s(i)+j}^{i} b_k(t)$$

for $1 \leq j \leq O-1$. They are related to the notation used in Kim et al. [17] according to $\beta_{i,j}(t) = \tilde{B}_{s(i)+j,O}(t)$ but only for $t \leq t_{i+1}$. Hence, they share for this t range also their derivatives:

$$\frac{\mathrm{d}}{\mathrm{d}t}\beta_{i,j}(t) = \frac{O-1}{t_{i+j} - t_{s(i)+j}} B_{s(i)+j,O-1}(t)$$

Note that the assumption $t \in [t_O, t_{I-O+1})$ (or equivalent $O \leq i \leq I - O$) is crucial for Eqs. (10) and (11) to hold in the given form. This assumption contrasts Kim et al. [17], where the first summand in Eq. (11) does not depend on t for any admissible t. For Eqs. (10) and (11) to hold without this assumption, the weight

$$\sum_{j=1}^{\min(i,I-O)} b_j(t)$$

is needed, and this weight is 1 if and only if $i \geq O$, given that $t \in [t_i, t_{i+1})$. The fact that this weight would prevent the B-spline from having the biequivariance property proven in Sec. III.E is an important reason for this assumption.

We will now generalize this form Eq. (11) to the Lie group $\mathcal{G}$, just as Kim et al. [17] do it for unit-length quaternions. Let $(\mathcal{G}, \cdot)$ be a connected finite-dimensional Lie group, with Lie algebra $\mathfrak{g}$. Let $\boldsymbol{\iota} \in \mathcal{G}$ denote its identity element. Instead of vectors $\boldsymbol{c}_i$, a Lie group-valued curve is defined by a tuple of Lie group elements as control vertices $\boldsymbol{g}_i \in \mathcal{G}$. The sum of vectors is naturally identified with the corresponding sequence of Lie group operations (written here as multiplication) in the order given by the indices. The vector difference $(\boldsymbol{c}_k - \boldsymbol{c}_{k-1})$, where $k = s(i) + j$, is generalized to the left fraction $(\boldsymbol{g}_{k-1}^{-1}\boldsymbol{g}_k)$. The scaling of these "differences" (by the $\beta_{i,j}(t)$) is done "in the Lie algebra" by exploiting the exponential and logarithm maps of $\mathcal{G}$. This way one gets the generalized form of Eq. (11) as

$$\boldsymbol{g}(t) \coloneqq \boldsymbol{g}_{s(i)} \prod_{j=1}^{O-1} \boldsymbol{r}_{i,j}(t) \tag{12a}$$

where

$$\boldsymbol{r}_{i,j}(t) \coloneqq \exp(\beta_{i,j}(t)\boldsymbol{\varphi}_{s(i)+j}), \quad \text{and} \quad \boldsymbol{\varphi}_k \coloneqq \log(\boldsymbol{g}_{k-1}^{-1}\boldsymbol{g}_k) \tag{12b}$$

This construction requires that the log function is defined on $\boldsymbol{g}_{k-1}^{-1}\boldsymbol{g}_k$. To guarantee this, one has to make sure that there are enough intermediate points to have all $\boldsymbol{g}_{k-1}^{-1}\boldsymbol{g}_k$ close enough to identity. In a connected Lie group, this should always be possible with finite many $\boldsymbol{g}_i$. Note that the vector space construction [Eq. (11)] is equivalent to the Lie group approach [Eq. (12)] when $(\mathcal{G}, \cdot)$ is chosen as the additive group $(\mathcal{V}, +)$ of the vector space $\mathcal{V}$, which is always a commutative Lie group. In that sense, these Lie group-valued B-splines are a true generalization of the traditional vector space-valued B-splines.

C. Theoretical Preparations

To facilitate the analysis of Lie group-valued B-splines, we will start with some theoretical preparations.

1. *Ambient Algebra Assumption*

Assumption 1: We assume the Lie group $(\mathcal{G}, \cdot)$ to be an embedded differential manifold of a unital associative $\mathbb{R}$-algebra $(\mathcal{A}, +, \cdot)$[¶] and a subgroup of $(\mathcal{A}_{\text{inv}}, \cdot)$, the group of multiplicative invertible elements in $\mathcal{A}$.

Assumption 1 allows us to take derivatives of curves through $\mathcal{G}$ in the ambient algebra $\mathcal{A}$, where we can easily use the product rule and benefit from a single global tangent space. All real matrix Lie groups, the unit-length quaternions, the unit-magnitude dual quaternions, and the unit-length complex numbers all naturally fulfill this assumption. Those who feel uncomfortable with the abstract concept of an $\mathbb{R}$-algebra are encouraged to think of the unit-length quaternions as $\mathcal{G}$ and the quaternions as $\mathcal{A}$, or the group SO(3) as $\mathcal{G}$ and $\mathbb{R}^{3\times3}$ as $\mathcal{A}$.

2. *Matrix Isomorphy*

Given a vector $\boldsymbol{a} \in \mathcal{A}$, we will use special operators to retrieve the matrices that represent the left, $\boldsymbol{a}^L$, or right, $\boldsymbol{a}^R$, multiplication by $\boldsymbol{a}$ (with respect to a fixed default basis for $\mathcal{A}$), that is, $\Psi(\boldsymbol{ab}) = \boldsymbol{a}^L\Psi(\boldsymbol{b}) = \boldsymbol{b}^R\Psi(\boldsymbol{a})$, for all $\boldsymbol{b} \in \mathcal{A}$, where $\Psi\colon \mathcal{A} \to \mathbb{R}^{\dim\mathcal{A}}$ maps the vectors to their coordinate tuples. In the following, we will implicitly identify $\mathcal{A}$ vectors with their coordinates. These operators, restricted to $\mathcal{G}$, are injective Lie group-homomorphism $(\cdot)^L$ and antihomomorphism[**] $(\cdot)^R$, from $\mathcal{G}$ into $\mathrm{GL}(\mathcal{A})$, the General Linear group over $\mathcal{A}$. Hence, these operators give us the opportunity to reduce computational complexity of Jacobian matrix evaluation by choosing to apply the multiplication to elements of $\mathcal{G}$ rather than $\mathcal{G}^L$ or $\mathcal{G}^R$. The mapping $(\cdot)^L$ also leads to the following theorem.

Theorem 1: $\mathcal{A}^L$ is a sub-$\mathbb{R}$-algebra of $\mathbb{R}^{\dim\mathcal{A}\times\dim\mathcal{A}}$, $\mathcal{G}^L$ is a matrix Lie group embedded in $\mathcal{A}^L$, and this pair is fully isomorphic to the pair of $\mathcal{G}$ and $\mathcal{A}$.

We will use this theorem to import knowledge about matrix Lie groups to our general setting. It also tells us that Assumption 1 restricts us, up to isomorphism, precisely to $\mathbb{R}$-matrix Lie groups. However, it is still of practical use to know that all derivations are correct in the nonmatrix examples, such as the unit-length quaternions. A proof is provided in Appendix D.

3. *Exponential Maps*

In both $\mathcal{A}$ and $\mathcal{G}$, there is a canonical definition of an exponential map. For the algebra $\mathcal{A}$, the exponential map $\exp_{\mathcal{A}}$ is defined by the

[¶]A unital associative algebra is an $\mathbb{R}$-vector space equipped with an $\mathbb{R}$-bilinear associative vector multiplication and a multiplicative identity element.

[**]The "anti-" prefix refers to the antihomomorphy identity, which one gets from the homomorphy identity by swapping the arguments of one of the binary operations.

usual power series. Applied to matrices, this yields the matrix exponential map. The exponential map $\exp_{\mathcal{G}}$ of a Lie group $\mathcal{G}$ is required to be smooth, and for all $\boldsymbol{v} \in \mathfrak{g}$, the directional exponential map $\exp^{\boldsymbol{v}}_{\mathcal{G}}\colon \mathbb{R} \to \mathcal{G},\ t \mapsto \exp_{\mathcal{G}}(t\boldsymbol{v})$ must fulfill two identities: the classical exponentiation identity $\exp^{\boldsymbol{v}}_{\mathcal{G}}(\alpha+\beta) = \exp^{\boldsymbol{v}}_{\mathcal{G}}(\alpha)\exp^{\boldsymbol{v}}_{\mathcal{G}}(\beta)$, for all $\alpha, \beta \in \mathbb{R}$, and the differential identity $\mathrm{d}/\mathrm{d}t \exp^{\boldsymbol{v}}_{\mathcal{G}}(t)|_{t=0} = \boldsymbol{v}$.

Theorem 2: In our setting, $\exp_{\mathcal{G}}$ is a restriction of $\exp_{\mathcal{A}}$ to $\mathfrak{g} \subset \mathcal{A}$, employing the natural identification of $\mathcal{G}$'s tangent spaces with subvector spaces of $\mathcal{A}$.

We exploit this property to simplify Jacobian calculation by interpreting our B-spline construction [Eq. (12)] as construction in $\mathcal{A}$. A proof is provided in Appendix E. From now on, we will not distinguish anymore formally between both exponential maps and write just exp.

D. Continuous-Time Batch State Estimation on Lie Groups

In Sec. III.B, we present a construction for Lie group-valued B-splines. We will now illustrate how they can be exploited to extend continuous-time batch state estimation (see Sec. III.A) to Lie groups.

To minimize the negative log-likelihood function L, as given in Eq. (9), but with $\boldsymbol{x}(t) \coloneqq \boldsymbol{g}(t, (\boldsymbol{g}_k)_{k=1}^K)$ given as Lie group-valued B-spline, it is necessary to perform a nonlinear minimization involving variables in the Lie group in question, namely the control vertices $(\boldsymbol{g}_k)_{k=1}^K \in \mathcal{G}$.

In the following formulation, we assume L to be given in the form $\|\boldsymbol{e}\|^2$, where $\boldsymbol{e} \in \mathbb{R}^\nu$, $\nu \in \mathbb{N}$. We call $\boldsymbol{e}$ the error function because it typically consists of the normalized residuals. To implement Gauss–Newton or Levenberg–Marquardt optimization algorithms to minimize the cost function L, we must have access to an expression for the Jacobian of $\boldsymbol{e}$ with respect to small changes in the estimated variables. For a estimated variable $\boldsymbol{g} \in \mathcal{G}$ of Lie group type (e.g., one of the control vertices), this requires a generalized Jacobian concept. It is usual to use a minimal perturbation, $\boldsymbol{\phi} \in \mathbb{R}^m$ (with $m = \dim \mathcal{G}$), in the Lie algebra, mapped to the Lie group via an exponential chart $\Phi_{\bar{\boldsymbol{g}}}$ centered at the current guess $\bar{\boldsymbol{g}}$

$$\boldsymbol{g}(\boldsymbol{\phi}) = \Phi_{\bar{g}}(\boldsymbol{\phi}) \coloneqq \exp(\mathcal{B}_{\mathfrak{g}}\boldsymbol{\phi})\bar{\boldsymbol{g}} \tag{13}$$

where $\mathcal{B}_{\mathfrak{g}}\boldsymbol{\phi} \coloneqq \sum_{k=1}^m \boldsymbol{\phi}_k \boldsymbol{b}_k$ denotes the vector in $\mathfrak{g}$ corresponding to the coordinates $\boldsymbol{\phi}$, with respect to $\mathcal{B}_{\mathfrak{g}}$, denoting the default basis for $\mathfrak{g}$. This yields the concept of a Jacobian for a function $\boldsymbol{e}\colon \mathcal{G} \to \mathbb{R}^\nu$ at $\bar{\boldsymbol{g}}$ in the local chart $\Phi_{\bar{\boldsymbol{g}}}$, which is defined to be the traditional Jacobian at $\boldsymbol{0} \in \mathbb{R}^m$ of the composed function $\boldsymbol{e} \circ \Phi_{\bar{\boldsymbol{g}}}\colon \mathbb{R}^m \to \mathbb{R}^\nu$. Unfortunately, although the concept of charts is well known from differential geometry, there seems to be no standard notation for these generalized Jacobians. We will denote it in this paper as $\partial \boldsymbol{e}/\partial \boldsymbol{\phi}$, while the local chart used is specified based on context, for the sake of brevity.

This generalized Jacobian is used like a normal Jacobian in a Gauss–Newton or Levenberg–Marquardt iteration, except that after it produces an answer for $\boldsymbol{\phi}$, the current guess is updated as $\bar{\boldsymbol{g}} \leftarrow \exp(\mathcal{B}_{\mathfrak{g}}\boldsymbol{\phi})\bar{\boldsymbol{g}}$. This update is constraint sensitive in that the updated $\bar{\boldsymbol{g}}$ is guarantied to not leave $\mathcal{G}$, even when a nonminimal representation such as a vector in $\mathcal{A}$ would theoretically allow such a departure.

An outline of the algorithm is provided in Algorithm 1. The umbrella variable $\bar{\boldsymbol{X}}$ comprises the current guesses for all the estimated variables. For example, the state B-spline's control vertices $(\boldsymbol{g}_k)_1^K$. In practice it can contain control vertices of multiple splines and other state variables.

The error function $\boldsymbol{e}(\bar{\boldsymbol{X}}, \boldsymbol{Z})$ will typically depend on the state B-spline via its value and derivatives, e.g., via the $\boldsymbol{h}_i$ in Eq. (2), and integrals along the trajectory, e.g., for Eq. (5). Therefore, we need analytical expression for the Lie group-valued B-spline's value (Sec. III.B), its time derivatives (Sec. III.F.1), a concept how to evaluate integrals (Sec. III.H), and generalized Jacobians (line 5 in Algorithm 1), for all involved expressions with respect to changes in its control vertices (Sec. III.F.2). To initialize the control vertices (line 3 in Algorithm 1), we also require an initialization strategy (Sec. III.G).

E. Biequivariance of This B-Spline Construction

In this section, we will prove that this B-spline construction has an important property for a Lie group-valued parameterized curve when used for physical modeling, called "bi-invariance" by Park and Ravani [62] for the special case of SO(3). From our point of view, the term biequivariance is a better fit because it requires the curve's value to transform the same way as its control vertices when transformed by left or right multiplication with a group element. The term "invariant" would indicate no change in the curve's value for the same operations, but that is neither the case nor intended for our splines or for the curves presented in [62]. In contrast equivariance of a map with respect to a group action is exactly that: the function's value changes according to the same group action as applied on its argument. First, we consider left-equivariance: equivariance with respect to left multiplication by any $\boldsymbol{u} \in \mathcal{A}_{\text{inv}}$, $L_u\colon \mathcal{A} \to \mathcal{A},\ \boldsymbol{a} \mapsto \boldsymbol{u}\boldsymbol{a}$. The expression for $\boldsymbol{r}_{i,j}$ in Eq. (12b) only depends on the control vertices through expressions like $\boldsymbol{g}_{k-1}^{-1}\boldsymbol{g}_k$. Such expressions are invariant with respect to L_u, rendering all the $\boldsymbol{r}$ invariant, too. It follows that

$$\boldsymbol{g}((L_u(\boldsymbol{g}_k))_1^K, t) = L_u(\boldsymbol{g}_{s(i)}) \prod_{j=1}^{O-1} \boldsymbol{r}_{i,j}(t) = L_u(\boldsymbol{g}((L_u(\boldsymbol{g}_k))_1^K, t)) \tag{14}$$

Next, we consider the right equivariance proof. For $\boldsymbol{u} \in \mathcal{A}_{\text{inv}}$, the conjugation by $\boldsymbol{u}$, $\pi_u\colon \mathcal{A} \to \mathcal{A},\ \boldsymbol{a} \mapsto \boldsymbol{u}\boldsymbol{a}\boldsymbol{u}^{-1}$ is an $\mathbb{R}$-algebra automorphism. Therefore, any (partial) function of type $\mathcal{A}^k \to \mathcal{A}$ for some $k \in \mathbb{N}$ is automatically equivariant with respect to conjugation by $\boldsymbol{u}$ if the function is $\mathcal{A}$-intrinsically defined. A function from $\mathcal{A}^k \to \mathcal{A}$ is called $\mathcal{A}$-intrinsically defined if and only if its definition is purely based on the structure of $\mathcal{A}$; informally this means that its value and its domain can be defined by only referring to its arguments, real numbers, $\boldsymbol{0}, \boldsymbol{1} \in \mathcal{A}$, the operations $+$, $*$, the limit in $\mathcal{A}$, the identity relation, set or logical operators, or other $\mathcal{A}$-intrinsically defined functions. This implies that the exp and log mappings on $\mathcal{A}$, and ultimately our whole B-spline construction [Eq. (12)], are equivariant with respect to conjugation by $\boldsymbol{u}$ when the time parameter is considered a constant real number. The B-spline's right-equivariance immediately follows from the simple fact that the right multiplication, $R_u\colon \mathcal{A} \to \mathcal{A},\ \boldsymbol{a} \mapsto \boldsymbol{a}\boldsymbol{u}$, is equal to a left multiplication after a conjugation, $R_u = L_u \circ \pi_{u^{-1}}$.

These equivariances hold for any $\boldsymbol{u} \in \mathcal{A}_{\text{inv}}$, but the transformed spline will remain $\mathcal{G}$-valued if and only if $\boldsymbol{u} \in \mathcal{G}$. Therefore, in practice, only the equivariances with respect to left or right multiplication by $\boldsymbol{u} \in \mathcal{G}$ will be of interest.

Algorithm 1 Continuous-time batch state estimation on Lie groups

1) $\boldsymbol{Z} \leftarrow$ GETMEASUREMENTS
2) $\boldsymbol{e} \leftarrow$ SETUPERRORFUNCTION $(\boldsymbol{Z})$
3) $\bar{\boldsymbol{X}} \leftarrow$ COMPUTEINITIALGUESSES $(\boldsymbol{e}, \boldsymbol{Z})$
4) **repeat** ▷ Minimize quadratic negative log likelihood function $\|\boldsymbol{e}\|^2$
5) $\quad \boldsymbol{J} \leftarrow$ COMPUTEJACOBIANS $(\boldsymbol{e}, \bar{\boldsymbol{X}})$
6) $\quad \boldsymbol{U} \leftarrow$ COMPUTEUPDATES $(\boldsymbol{e}, \bar{\boldsymbol{X}}, \boldsymbol{J}, \boldsymbol{Z})$ ▷ Gauss–Newton or Levenberg–Marquardt
7) $\quad \bar{\boldsymbol{X}} \leftarrow \exp(\boldsymbol{U})\bar{\boldsymbol{X}}$
8) **until** HASCONVERGED $(\boldsymbol{e}, \bar{\boldsymbol{X}}, \boldsymbol{Z})$

F. Derivatives

1. Time Derivatives

We can calculate the B-splines derivatives with respect to t using the usual product rule supported by Assumption 1, which allows us to interpret the whole B-spline construction [Eq. (12)] as defined more generally in $\mathcal{A}$:

$$\frac{\mathrm{d}^k}{\mathrm{d}t^k}\boldsymbol{g}(t)=\frac{\mathrm{d}^k}{\mathrm{d}t^k}\boldsymbol{g}_{s(i)}\prod_{j=1}^{O-1}\boldsymbol{r}_{i,j}(t)=k!\boldsymbol{g}_{s(i)}\sum_{\substack{\alpha\in\mathbb{N}^{O-1},\\ \sum\alpha=k}}\prod_{j=1}^{O-1}\frac{1}{\boldsymbol{\alpha}_j!}\frac{\mathrm{d}^{\alpha_j}}{\mathrm{d}t^{\alpha_j}}\boldsymbol{r}_{i,j}(t) \quad (15)$$

To be complete, it requires the time derivatives of $\boldsymbol{r}_{i,j}$ up to order k. We will exploit the directional exponential map's well known differential identity (guaranteed in $\mathcal{A}$ by Theorem 1):

$$\frac{\mathrm{d}}{\mathrm{d}t}\exp^a(t)=\boldsymbol{a}\exp^a(t),\quad \text{where } \exp^a(t)\coloneqq\exp(t\boldsymbol{a}) \quad (16)$$

where $\boldsymbol{a}\in\mathcal{A}$ and $t\in\mathbb{R}$.

Using Theorem 2 to identify the exp and log in Eq. (12b) with their extensions to $\mathcal{A}$, we get, with $k\coloneqq s(i)+j$,

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{r}_{i,j}(t)&=\frac{\mathrm{d}}{\mathrm{d}t}\exp^{\boldsymbol{\varphi}_k}(\beta_{i,j}(t))=\beta'_{i,j}(t)\boldsymbol{\varphi}_k\exp^{\boldsymbol{\varphi}_k}(\beta_{i,j}(t))\\&=\beta'_{i,j}(t)\boldsymbol{\varphi}_k\boldsymbol{r}_{i,j}(t)\end{aligned} \quad (17a)$$

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\boldsymbol{r}_{i,j}(t)=(\beta'_{i,j}(t)^2\boldsymbol{\varphi}_k+\beta''_{i,j}(t)\boldsymbol{\imath})\boldsymbol{\varphi}_k\boldsymbol{r}_{i,j}(t) \quad (17b)$$

$$\begin{aligned}\frac{\mathrm{d}^3}{\mathrm{d}t^3}\boldsymbol{r}_{i,j}(t)=&\left(\beta'_{i,j}(t)\boldsymbol{\varphi}_k(\beta'_{i,j}(t)^2\boldsymbol{\varphi}_k+\beta''_{i,j}(t)+2\beta'_{i,j}(t)\beta''_{i,j}(t)\boldsymbol{\varphi}_k+\beta'''_{i,j}(t)\boldsymbol{\imath}\right)\\&\cdot\boldsymbol{\varphi}_k\boldsymbol{r}_{i,j}(t)\end{aligned} \quad (17c)$$

We will not need higher-order derivatives for our experiments. For how to compute the β derivatives, see our comment after Eq. (11) and in de Boor [61].

2. Jacobians with Respect to Changes in the Control Vertices

In this section, we will derive the partial Jacobians of the B-spline's value and derivatives with respect to small changes, $\boldsymbol{\phi}_l$, while considering the control vertices, $\boldsymbol{g}_l(\boldsymbol{\phi}_l)\coloneqq\Phi_{\bar{g}_l}(\boldsymbol{\phi})=\exp(\mathcal{B}_{\mathfrak{g}}\boldsymbol{\phi})\bar{\boldsymbol{g}}_l$, as functions of $\boldsymbol{\phi}_k$, as explained in Sec. III.D. Because the chart map $\Phi_{\bar{g}}$ is centered at $\bar{\boldsymbol{g}}$, all Jacobians will be implicitly at $0\in\mathbb{R}^m$. The value $\in\mathcal{G}\subset\mathcal{A}$ and time derivatives $\in\mathcal{A}$ of the B-spline will be implicitly identified with their coordinates with respect to $\mathcal{A}$'s default basis.

Omitting the time parameter we get for $l\in\{1\dots K\}$,

$$\begin{aligned}\frac{\partial\boldsymbol{g}}{\partial\boldsymbol{\phi}_l}=&\frac{\partial}{\partial\boldsymbol{\phi}_l}\boldsymbol{g}_{s(i)}\prod_{j=1}^{O-1}\boldsymbol{r}_{i,j}=\left(\prod_{j=1}^{O-1}\boldsymbol{r}_{i,j}\right)^R\frac{\partial\boldsymbol{g}_{s(i)}}{\partial\boldsymbol{\phi}_l}\\&+\boldsymbol{g}^L_{s(i)}\sum_{\hat{j}=1}^{O-1}\left(\prod_{j=1}^{\hat{j}-1}\boldsymbol{r}_{i,j}\right)^L\left(\prod_{j=\hat{j}+1}^{O-1}\boldsymbol{r}_{i,j}\right)^R\frac{\partial\boldsymbol{r}_{i,\hat{j}}}{\partial\boldsymbol{\phi}_l}\end{aligned} \quad (18)$$

For the first summand, we get

$$\begin{aligned}\frac{\partial\boldsymbol{g}_{s(i)}}{\partial\boldsymbol{\phi}_l}&=\frac{\partial}{\partial\boldsymbol{\phi}_l}\exp(\mathcal{B}_{\mathfrak{g}}\boldsymbol{\phi}_{s(i)})\bar{\boldsymbol{g}}_{s(i)}=\bar{\boldsymbol{g}}^R_{s(i)}\frac{\partial}{\partial\boldsymbol{\phi}_l}\exp(\mathcal{B}_{\mathfrak{g}}\boldsymbol{\phi}_{s(i)})\\&=\delta_{l,s(i)}\bar{\boldsymbol{g}}^R_{s(i)}\boldsymbol{V}\end{aligned} \quad (19)$$

where δ is the Kronecker delta, and $\boldsymbol{V}$ is the Jacobian matrix of $\mathcal{G}$'s exponential map in coordinates at $\boldsymbol{0}$. Note that, because the differential of the exp map in $\mathcal{A}$ at zero is the identity map, this $\boldsymbol{V}$ matrix is also the Jacobian of the implicit identification $\mathfrak{g}$ into $\mathcal{A}$. We can use $\boldsymbol{V}$ to convert coordinates with respect to $\mathcal{B}_{\mathfrak{g}}$ to coordinates in $\mathcal{A}$.

Next, we derive the Jacobian $(\partial/\partial\boldsymbol{\phi}_l)\boldsymbol{r}_{i,j}$, for any $1\le j<O$ and with $k\coloneqq s(i)+j$. To begin, we will use the exponential chart to decompose $\boldsymbol{r}_{i,j}$ into a chain of maps between vector spaces allowing application of the traditional chain rule:

$$\begin{aligned}\boldsymbol{r}_{i,j}&=\exp(\beta_{i,j}\boldsymbol{\varphi}_k)=\exp(\beta_{i,j}\log(\boldsymbol{g}_{k-1}^{-1}\boldsymbol{g}_k))\\&=\underbrace{\exp}_{\mathrm{I}}\circ(\beta_{i,j}\cdot)\circ\underbrace{(\log\circ\Phi_{\bar{d}})}_{\mathrm{II}}\circ\underbrace{(\Phi_{\bar{d}}^{-1}\circ\boldsymbol{d})}_{\mathrm{III}}\end{aligned} \quad (20)$$

with $\boldsymbol{d}\coloneqq\boldsymbol{g}_{k-1}^{-1}(\boldsymbol{\phi}_{k-1})\boldsymbol{g}_k(\boldsymbol{\phi}_k)$, and $\bar{\boldsymbol{d}}=\bar{\boldsymbol{g}}_{k-1}^{-1}\bar{\boldsymbol{g}}_k$.

The Jacobian for I, $\mathcal{G}$'s exponential map at $\bar{\boldsymbol{v}}\coloneqq\beta_{i,j}\bar{\boldsymbol{\varphi}}_k$, is usually known for the specific $\mathcal{G}$. We label it with $\boldsymbol{E}(\bar{\boldsymbol{v}})$ and assume it to be given. If it is instead available for the exponential map in exponential chart, $\Phi^{-1}_{\exp(v)}\circ\exp$, here denoted with $\boldsymbol{S}(\bar{\boldsymbol{v}})$, one can use the identity $\boldsymbol{E}=\exp(\bar{\boldsymbol{v}})^R\boldsymbol{V}\boldsymbol{S}(\bar{\boldsymbol{v}})$ to retrieve it. It holds because $\exp(\bar{\boldsymbol{v}})^R\boldsymbol{V}$ is the Jacobian of $\Phi_{\exp(\bar{v})}$ at zero. In case none is available, a general formula for this $\boldsymbol{S}$ matrix is well known ([59] p. 70]):

$$\boldsymbol{S}(\boldsymbol{v})=\sum_{k=0}^{\infty}\frac{(-1)^k}{(k+1)!}(\mathrm{ad}_v)^k \quad (21)$$

where ad_v denotes the matrix representing the adjoint action $\boldsymbol{w}\mapsto[\boldsymbol{v},\boldsymbol{w}]$, where the Lie bracket $[\boldsymbol{v},\boldsymbol{w}]$ equals $\boldsymbol{v}\boldsymbol{w}-\boldsymbol{w}\boldsymbol{v}$, using the algebra multiplication, in our assumed setting.

The Jacobian of II, the logarithm in the exponential chart, $\log\circ\Phi_{\bar{d}}$, we label with $\boldsymbol{L}(\bar{\boldsymbol{d}})$ and assume it to be given as well. If not, it can be retrieved as the inverse of $\boldsymbol{S}(\log(\bar{\boldsymbol{d}}))$ because $\log\circ\Phi_d\circ\Phi^{-1}_{\exp(\log(\bar{d}))}\circ\exp=\mathrm{Id}$.

The last piece is the Jacobian of III:

$$\frac{\partial}{\partial\boldsymbol{\phi}_l}\Phi_d^{-1}(\boldsymbol{d})=(\delta_{l,k}-\delta_{l,k-1})\boldsymbol{C}(\bar{\boldsymbol{g}}_{k-1}^{-1}) \quad (22)$$

as proven in Appendix C, with

$$\boldsymbol{C}(\boldsymbol{g})\coloneqq\boldsymbol{W}\boldsymbol{g}^L\boldsymbol{g}^R\boldsymbol{V} \quad (23)$$

the Jacobian at identity of the adjoint operation of $\boldsymbol{g}$ on $\mathfrak{g}$, where $\boldsymbol{W}$ denotes the Jacobian of the log at $\boldsymbol{\imath}$.

Altogether, we have

$$\frac{\partial\boldsymbol{r}_{i,j}}{\partial\boldsymbol{\phi}_l}=\boldsymbol{E}(\beta_{i,j}\bar{\boldsymbol{\varphi}}_k)\beta_{i,j}\frac{\partial\boldsymbol{\varphi}_k}{\partial\boldsymbol{\phi}_l}=\bar{\boldsymbol{r}}^R_{i,j}\boldsymbol{V}\boldsymbol{S}(\beta_{i,j}\bar{\boldsymbol{\varphi}}_k)\beta_{i,j}\frac{\partial\boldsymbol{\varphi}_k}{\partial\boldsymbol{\phi}_l} \quad (24a)$$

with

$$\frac{\partial\boldsymbol{\varphi}_k}{\partial\boldsymbol{\phi}_l}=(\delta_{l,k}-\delta_{l,k-1})\boldsymbol{L}(\bar{\boldsymbol{g}}_{k-1}^{-1}\bar{\boldsymbol{g}}_k)\boldsymbol{C}(\bar{\boldsymbol{g}}_{k-1}^{-1}) \quad (24b)$$

According to Sec. III.D, we also need the corresponding Jacobians for the time derivatives of the B-spline. Starting with Eq. (15) and applying the same method as for Eq. (18), we have

$$\frac{\partial}{\partial\boldsymbol{\phi}_l}\frac{\mathrm{d}^k}{\mathrm{d}t^k}\boldsymbol{g}=k!\frac{\partial}{\partial\boldsymbol{\phi}_l}\boldsymbol{g}_{s(i)}\boldsymbol{a}_{k,1,O} \quad (25a)$$

$$=k!\boldsymbol{a}_{k,1,O-1}\frac{\partial\boldsymbol{g}_{s(i)}}{\partial\boldsymbol{\phi}_l}+k!\boldsymbol{g}^L_{s(i)}\sum_{\hat{j}=1}^{O-1}\sum_{\substack{\gamma\in\mathbb{N}^3,\\ \sum\gamma=k}}\boldsymbol{a}^L_{\gamma_1,1,j-1}\boldsymbol{a}^R_{\gamma_2,j+1,O}\frac{\partial}{\partial\boldsymbol{\phi}_l}D^{\gamma_3}\boldsymbol{r}_{i,\hat{j}} \quad (25b)$$

$$= k!\boldsymbol{a}^R_{k,1,O-1}\frac{\partial \boldsymbol{g}_{s(i)}}{\partial \boldsymbol{\phi}_l} + k!\boldsymbol{g}^L_{s(i)}\sum_{\hat{j}=1}^{O-1}\sum_{\gamma_1=0}^{k}\boldsymbol{a}^L_{\gamma_1,1,j-1}\left(\sum_{\gamma_2=0}^{k-\gamma_1}\boldsymbol{a}^R_{\gamma_2,j+1,O}\frac{\partial}{\partial \boldsymbol{\phi}_l}D^{k-\gamma_1-\gamma_2}\boldsymbol{r}_{i,\hat{j}}\right) \quad (25c)$$

with

$$\boldsymbol{a}_{k,j_F,j_L} \coloneqq \sum_{\substack{\alpha\in\mathbb{N}^{j_L-j_F},\\ \sum\alpha=k}}\prod_{j=j_F}^{j_L-1}D^{\alpha_{j-j_F+1}}\boldsymbol{r}_{i,j},$$

$$\text{if } j_L-1>j_F \text{ else } \iota, \text{ and } D^\mu \coloneqq \frac{1}{\mu!}\frac{\mathrm{d}^\mu}{\mathrm{d}t^\mu} \quad (25d)$$

The last transformation [Eq. (25c)] is only to reduce computational complexity by exploiting the distributive law of matrix multiplication. Combining Eqs. (17) and (24), we have (not expanding the derivatives of $\boldsymbol{\varphi}_k$ and only up to $k=2$ for the sake of readability):

$$\frac{\partial}{\partial \boldsymbol{\phi}_l}\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{r}_{i,j}(t) = \beta'_{i,j}(t)\frac{\partial}{\partial \boldsymbol{\phi}_l}(\boldsymbol{V}\boldsymbol{\varphi}_k)\boldsymbol{r}_{i,j}(t) = \beta'_{i,j}(t)\left(\boldsymbol{r}_{i,j}(t)^R\boldsymbol{V}\frac{\partial \boldsymbol{\varphi}_k}{\partial \boldsymbol{\phi}_l} + (\boldsymbol{V}\boldsymbol{\varphi}_k)^L\frac{\partial \boldsymbol{r}_{i,j}(t)}{\partial \boldsymbol{\phi}_l}\right) \quad (26a)$$

$$\frac{\partial}{\partial \boldsymbol{\phi}_l}\frac{\mathrm{d}^2}{\mathrm{d}t^2}\boldsymbol{r}_{i,j}(t) = (\beta'_{i,j}(t)^2\boldsymbol{V}\boldsymbol{\varphi}_k + \beta''_{i,j}(t)\boldsymbol{\iota})^L \times\left(\boldsymbol{r}_{i,j}(t)^R\boldsymbol{V}\frac{\partial \boldsymbol{\varphi}_k}{\partial \boldsymbol{\phi}_l} + (\boldsymbol{V}\boldsymbol{\varphi}_k)^L\frac{\partial \boldsymbol{r}_{i,j}(t)}{\partial \boldsymbol{\phi}_l}\right) + ((\boldsymbol{V}\boldsymbol{\varphi}_k)\boldsymbol{r}_{i,j}(t))^R\left(\beta'_{i,j}(t)^2\boldsymbol{V}\frac{\partial \boldsymbol{\varphi}_k}{\partial \boldsymbol{\phi}_l}\right) \quad (26b)$$

G. Rough B-Spline Fitting for Initialization

To initialize the nonlinear optimization of a $\mathcal{G}$-valued curve, we will need a strategy to come to a suitable initial guess. Formally, this is the problem of fitting a curve defined on $\mathcal{T}=[0,T]$ to a time series in $\mathcal{G}$, $\alpha \coloneqq (\tau_k,\boldsymbol{g}_k)_{k=1}^K \in \mathcal{T}\times\mathcal{G}$, while somehow penalizing high acceleration, and preventing overfitting. For our experiments, we used the following strategy to get a rough fit of our B-spline by making use of Assumption 1. Because we have $\boldsymbol{g}(t,(\boldsymbol{g}_k)_{k=1}^K)\in\mathcal{G}\subset\mathcal{A}_{\text{inv}}\subset\mathcal{A}$ for all times $t\in\mathcal{T}$ and control vertices $\boldsymbol{g}_k\in\mathcal{G}$, we can consider the whole $\mathcal{G}$-valued B-spline construct as a Lie group B-spline in the Lie group $(\mathcal{A}_{\text{inv}},\cdot)$, i.e., based on the multiplication in $\mathcal{A}$. In $\mathcal{A}$, we also have the $\mathbb{R}$-vector space structure and we can define for each multiplicative $\boldsymbol{g}(t,(\boldsymbol{g}_k)_{k=1}^K)$ the corresponding additive B-spline $\boldsymbol{g}^{\text{add}}(t,(\boldsymbol{g}_k)_{k=1}^K)$ as defined in the additive Lie group, $(\mathcal{A},+)$. These are now traditional vector space-valued B-splines, and we know how to efficiently fit them to the time series α. This is a linear problem, even with the acceleration penalty as regularization (see the solution of Schoenberg and Reinsch as described in Chap. 14 of [61]). After the linear fit, we may need to modify the control vertices to make them elements in $\mathcal{G}$. As long as they are not too far off, this can be done uniquely and, for some common examples, also efficiently. For example in the case of unit-length quaternions, this just means normalizing the resulting vectors in $\mathbb{R}^4$ to unit length.

To summarize, we get our initial guess for optimization using the following steps.

1) Acquire control vertices $\boldsymbol{g}_k\in\mathcal{A}$ by doing a least-squares fit of $\boldsymbol{g}^{\text{add}}$ to α while keeping the integral over the spline's acceleration low (by introducing an appropriate, linear cost scaled with an acceleration penalty factor).

2) Project the $\boldsymbol{g}_k$ into $\mathcal{G}$.

3) Use the modified $\boldsymbol{g}_k$ as control vertices for a multiplicative spline in $\mathcal{G}$.

Despite being very simple, this method worked extremely well in all of our experiments up to B-spline order 12, where the linear solution for the initialization fell into local likelihood maxima for tight knot spacing when using our default acceleration penalty. This could be solved by rising the acceleration penalty factor from 10^4 to 10^6. Automatic tuning of this parameter is out of the scope of this paper.

Note that it is not equally good to just use a subset of the values in α as control vertices, even though this would also result in a somewhat close to curve. This would leave us no means to penalize high acceleration, which would allow outliers to badly affect the result.

H. Integrals Along Curves

To deal with cost function (summands) that involve integrals along our B-splines (e.g., to impose dynamic system models), we used the following approach. Given a time interval $\mathcal{T}=[0,T]$ and a cost functional J defined on continuous curves through $\mathcal{G}$, $\gamma\colon\mathcal{T}\to\mathcal{G}$ such that

$$J(\gamma)=\int_0^T f(\gamma(t))^T W(t) f(\gamma(t))\,\mathrm{d}t$$

where $l\in\mathbb{N}$, $W\colon\mathcal{T}\to\mathbb{R}^{l\times l}$ and $f\colon\mathcal{G}\to\mathbb{R}^l$, each also continuous, we apply a nonadaptive numeric integration scheme $(w,\xi)\colon\{1\dots k\}\to\mathbb{R}\times\mathcal{T}$ (e.g., Simpson's, Trapezoidal rule, etc.), such that

$$\underset{g\in\mathcal{C}(\mathcal{T},\mathbb{R})}{\forall}\quad \sum_{i=1}^k w_i g(\xi_i)\approx\int_0^T g(t)\,\mathrm{d}t$$

to convert J into a sequence of quadratic error terms (to interface a Gauss–Newton optimization package), this becomes

$$J(\gamma)\approx\sum_{i=1}^k w_i f(\gamma(\xi_i))^T W(\xi_i) f(\gamma(\xi_i))$$

I. Unit-Length Quaternions Lie Group

In this section, we will focus on special aspects for the Lie group of unit-length quaternions when used as representative for SO(3). To be prepared for that, we will shortly define some notation. First, let $\mathbb{H}$ denote the skew field and $\mathbb{R}$-algebra of quaternions and

$$\mathbb{S}^3 \coloneqq \{\boldsymbol{q}\in\mathbb{H}\,|\,|\boldsymbol{q}|=1\}$$

denote the three-dimensional $\mathbb{R}$-Lie group of unit-length quaternions. We can make use of the theory in Sec. III.C using the following identification. In this setting, $\mathbb{H}$ is the the ambient $\mathbb{R}$-algebra $(\mathcal{A})$, and $\mathbb{S}^3$ is the embedded Lie group $(\mathcal{G})$.

We will rely on the notation presented in Barfoot et al. [63]. Let $\boldsymbol{\iota}$ denote the identity element of $\mathbb{S}^3$, which is equal to the multiplicative identity element in $\mathbb{H}$. By using $(\boldsymbol{i},\boldsymbol{j},\boldsymbol{k},\boldsymbol{\iota})$ as a basis for $\mathbb{H}$, we can write the coordinates of a quaternion $\boldsymbol{q}$ as

$$\boldsymbol{q} =: \begin{bmatrix}\boldsymbol{\epsilon}\\ \eta\end{bmatrix} \quad (27)$$

where $\boldsymbol{\epsilon}$ is 3×1, and η is a scalar. We will denote the product of two quaternions $\boldsymbol{p},\boldsymbol{q}\in\mathbb{H}$ with $\boldsymbol{p}\cdot\boldsymbol{q}$ or just $\boldsymbol{p}\boldsymbol{q}$ and assume Hamilton's multiplication order (i.e., $\boldsymbol{i}\boldsymbol{j}=\boldsymbol{k}$ holds). The left and right multiplication matrices, $(\cdot)^L$ and $(\cdot)^R$, are

$$\boldsymbol{q}^L=\begin{bmatrix}\eta\mathbf{1}+\boldsymbol{\epsilon}^\times & \boldsymbol{\epsilon}\\ -\boldsymbol{\epsilon}^T & \eta\end{bmatrix}\quad\text{and}\quad \boldsymbol{q}^R=\begin{bmatrix}\eta\mathbf{1}-\boldsymbol{\epsilon}^\times & \boldsymbol{\epsilon}\\ -\boldsymbol{\epsilon}^T & \eta\end{bmatrix} \quad (28)$$

where $\mathbf{1}$ is the 3×3 identity matrix, and $\boldsymbol{\epsilon}^\times$ denotes the matrix such that for all $x\in\mathbb{R}^3$, $\boldsymbol{\epsilon}\times x=\boldsymbol{\epsilon}^\times x$. The quaternion compound

operators, $(\cdot)^{\oplus}$, $(\cdot)^{+}$ of Barfoot et al. [63], are precisely $(\cdot)^{L}$ and $(\cdot)^{R}$. Given a unit-length quaternion $\boldsymbol{q} \in \mathbb{S}^3$, the complex conjugate operator coincides with the multiplicative inverse operator:

$$\boldsymbol{q}^{-1} = \begin{bmatrix} -\boldsymbol{\epsilon} \\ \eta \end{bmatrix} \tag{29}$$

The Lie algebra of $\mathbb{S}^3$ consists exactly of the pure imaginary quaternions. We pick the usual basis $\mathcal{B}_{\mathfrak{g}} \coloneqq (\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k})$ and represent this Lie algebra's vectors with coordinate vectors $\boldsymbol{\phi} \in \mathbb{R}^3$. To convert from three to four-vectors and back, we use the matrices

$$\boldsymbol{V} = \begin{bmatrix} \boldsymbol{1} \\ \boldsymbol{0}^T \end{bmatrix}, \qquad \text{implying } \boldsymbol{V}\boldsymbol{\phi} = \begin{bmatrix} \boldsymbol{\phi} \\ 0 \end{bmatrix},$$
$$\text{and} \quad \boldsymbol{W} = \boldsymbol{V}^T, \qquad \text{implying } \boldsymbol{W}\boldsymbol{V} = \boldsymbol{1} \tag{30}$$

which correspond to the matrices introduced after Eqs. (19) and (23), respectively, as the Jacobians of the exponential map, at zero, and logarithm map, at identity.

We will identify a unit quaternion $\boldsymbol{q}$ with the formal rotation

$$\mathrm{Rot}_q \colon \mathbb{R}^3 \to \mathbb{R}^3, \qquad \boldsymbol{x} \mapsto \boldsymbol{W}(\boldsymbol{q}(\boldsymbol{V}\boldsymbol{x})\boldsymbol{q}^{-1}) \tag{31}$$

The proper orthogonal matrix $\boldsymbol{C} \in \mathrm{SO}(3)$ associated with the same rotation by $\forall_{x\in\mathbb{R}^3}\ \boldsymbol{C}\boldsymbol{x} = \mathrm{Rot}_q(\boldsymbol{x})$ may be computed using

$$\boldsymbol{C} = \boldsymbol{W}\boldsymbol{q}^{L}\boldsymbol{q}^{-1R}\boldsymbol{V} \tag{32}$$

This is precisely the matrix $\boldsymbol{C}$ defined in Eq. (23).

This unit-length quaternion's log and exp functions can be calculated in the following way:

$$\exp(\boldsymbol{\phi}) = \begin{cases} \boldsymbol{\iota}, & \boldsymbol{\phi} = \boldsymbol{0} \\ \begin{bmatrix} \sin(\phi)\boldsymbol{a} \\ \cos\phi \end{bmatrix}, & \boldsymbol{\phi} \neq \boldsymbol{0} \end{cases}, \quad \log(\boldsymbol{q}) = \begin{cases} \boldsymbol{0}, & \boldsymbol{q} = \boldsymbol{\iota} \\ \frac{\arccos\eta}{\sqrt{1-\eta^2}}\boldsymbol{\epsilon}, & \boldsymbol{q} \neq \pm\boldsymbol{\iota} \\ \text{undefined}, & \boldsymbol{q} = -\boldsymbol{\iota} \end{cases} \tag{33}$$

where $\boldsymbol{\phi}$ and the log's value are 3×1 coordinate vectors with respect to $\mathcal{B}_{\mathfrak{g}}$, $\phi \coloneqq \|\boldsymbol{\phi}\|$, and $\boldsymbol{a} \coloneqq \boldsymbol{\phi}/\phi$. A derivation of Eq. (33) can be found in the appendix of Kim and Nam [64].

J. Derivatives of Quaternion B-Splines

In this section, we give the time derivative and Jacobian formulas specific to our formulation for curves over $\mathbb{S}^3$. The B-spline time derivatives we calculated for the general case in Eq. (17) apply immediately to a unit-length quaternion curve, $\boldsymbol{q}(t) = [\boldsymbol{\epsilon}(t)^T \eta(t)]^T$, giving us directly quantities like $\dot{\boldsymbol{\epsilon}}(t)$, $\ddot{\boldsymbol{\epsilon}}(t)$, $\dot{\eta}(t)$, $\ddot{\eta}(t)$, etc. However, when we use such a curve to represent rotations as part of a physical model, we also need expressions for the angular velocity $\boldsymbol{\omega}$ and angular acceleration $\boldsymbol{\alpha}$ of a rotating frame.

To derive formulas for $\boldsymbol{\omega}$ or $\boldsymbol{\alpha}$, as physical notions, we need to specify how we interpret the formal rotation [Eq. (31)] in a physical context. We will identify the rotation from frame $\underset{\rightarrow a}{\mathcal{F}}$, an inertial frame, to $\underset{\rightarrow b}{\mathcal{F}}$, one attached to a rigid body, by the unit-length quaternion $\boldsymbol{q}_{ba}$, such that

$$\forall_{v\in\mathbb{E}^3}\ \ \mathrm{Rot}_{\boldsymbol{q}_{ba}}(\boldsymbol{v}_a) = \boldsymbol{v}_b \tag{34}$$

where $\boldsymbol{v}_a$ and $\boldsymbol{v}_b$ denote the coordinates of a vector $\boldsymbol{v}$ in Euclidean space ($\mathbb{E}$) with respect to $\underset{\rightarrow a}{\mathcal{F}}$ and $\underset{\rightarrow b}{\mathcal{F}}$, respectively. The quaternion $\boldsymbol{q}_{ba}$ is only uniquely defined up to negation, and thus all physical equations need to be invariant with respect to negation of this variable.

1. Angular Velocity

For the angular velocity $\boldsymbol{\omega}$ of $\underset{\rightarrow b}{\mathcal{F}}$ with respect to $\underset{\rightarrow a}{\mathcal{F}}$ given by a quaternion-valued curve through time $\boldsymbol{q}_{ba}(t) = [\boldsymbol{\epsilon}(t)^T \eta(t)]^T$, we get (omitting the time parameter for clarity):

$$\boldsymbol{\omega}_a = -2\boldsymbol{W}(\boldsymbol{q}_{ba}^{-1}\dot{\boldsymbol{q}}_{ba}) = -2((\eta\dot{\boldsymbol{\epsilon}} - \dot{\eta}\boldsymbol{\epsilon}) - \boldsymbol{\epsilon}^{\times}\dot{\boldsymbol{\epsilon}}) \tag{35}$$

when expressed in $\underset{\rightarrow a}{\mathcal{F}}$, and

$$\boldsymbol{\omega}_b = -2\boldsymbol{W}(\dot{\boldsymbol{q}}_{ba}\boldsymbol{q}_{ba}^{-1}) = -2((\eta\dot{\boldsymbol{\epsilon}} - \dot{\eta}\boldsymbol{\epsilon}) - \dot{\boldsymbol{\epsilon}}^{\times}\boldsymbol{\epsilon}) \tag{36}$$

when expressed in $\underset{\rightarrow b}{\mathcal{F}}$. Both equations are proven in Appendix A.

2. Angular Acceleration

It follows for the angular acceleration α, omitting the subscripts for $\boldsymbol{q}_{ba}$ for brevity:

$$\alpha_b = \dot{\boldsymbol{\omega}}_b = -2\boldsymbol{W}\left(\frac{\mathrm{d}}{\mathrm{d}t}(\dot{\boldsymbol{q}}\boldsymbol{q}^{-1})\right) = -2\boldsymbol{W}(\dot{\boldsymbol{q}}^{-1}\dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}}\boldsymbol{q}^{-1}) = -2\boldsymbol{W}(\ddot{\boldsymbol{q}}\boldsymbol{q}^{-1}) \tag{37}$$

$$= -2((\eta\ddot{\boldsymbol{\epsilon}} - \ddot{\eta}\boldsymbol{\epsilon}) - \ddot{\boldsymbol{\epsilon}}^{\times}\boldsymbol{\epsilon}) \tag{38}$$

3. Jacobians of Quaternion B-Splines with Respect to the Control Vertices

To use the results from Sec. III.F.2, we will need the specific matrices for the unit-length quaternions. The Jacobian of the exponential map, $\boldsymbol{S}(\boldsymbol{\phi})$, and of the logarithm map, $\boldsymbol{L}(\boldsymbol{q})$, using the same variables as in Eq. (33), additionally assuming for $\boldsymbol{L}(\boldsymbol{q})$ that $\log(\boldsymbol{q}) = \boldsymbol{\phi}$. Both are $\boldsymbol{1} \in \mathbb{R}^{3\times3}$ in case $\boldsymbol{\phi} = 0$ and $\boldsymbol{q} = \boldsymbol{\iota}$, respectively. Otherwise,

$$\boldsymbol{S}(\boldsymbol{\phi}) = \boldsymbol{1} - \frac{1}{\phi}\sin^2\phi\boldsymbol{a}^{\times} + \left(1 - \frac{1}{2\phi}\sin 2\phi\right)\boldsymbol{a}^{\times}\boldsymbol{a}^{\times} \tag{39}$$

and

$$\boldsymbol{L}(\boldsymbol{q}) = \boldsymbol{1} + \boldsymbol{\phi}^{\times} + \left(1 - \frac{\phi}{\tan\phi}\right)\boldsymbol{a}^{\times}\boldsymbol{a}^{\times} \quad \text{for } \boldsymbol{q} \neq \pm\boldsymbol{\iota} \tag{40}$$

These formulas are proven/derived in Appendix B.

4. Jacobians of Angular Velocity

The Jacobians of the angular velocity with respect to minimal perturbations of the control vertices are also required. For $1 \leq j < O$ and $l \in \{1 \ldots K\}$, we get

$$\frac{\partial\boldsymbol{\omega}_a}{\partial\boldsymbol{\phi}_l} = -2\left(\frac{\partial\eta}{\partial\boldsymbol{\phi}_l}\dot{\boldsymbol{\epsilon}} + \eta\frac{\partial\dot{\boldsymbol{\epsilon}}}{\partial\boldsymbol{\phi}_l} - \frac{\partial\dot{\eta}}{\partial\boldsymbol{\phi}_l}\boldsymbol{\epsilon} - \dot{\eta}\frac{\partial\boldsymbol{\epsilon}}{\partial\boldsymbol{\phi}_l} + \dot{\boldsymbol{\epsilon}}^{\times}\frac{\partial\boldsymbol{\epsilon}}{\partial\boldsymbol{\phi}_l} - \boldsymbol{\epsilon}^{\times}\frac{\partial\dot{\boldsymbol{\epsilon}}}{\partial\boldsymbol{\phi}_l}\right) \tag{41}$$

5. Jacobians of Angular Acceleration

For the angular acceleration, it yields analogously

$$\frac{\partial\boldsymbol{\alpha}_a}{\partial\boldsymbol{\phi}_l} = -2\left(\frac{\partial\eta}{\partial\boldsymbol{\phi}_l}\ddot{\boldsymbol{\epsilon}} + \eta\frac{\partial\ddot{\boldsymbol{\epsilon}}}{\partial\boldsymbol{\phi}_l} - \frac{\partial\ddot{\eta}}{\partial\boldsymbol{\phi}_l}\boldsymbol{\epsilon} - \ddot{\eta}\frac{\partial\boldsymbol{\epsilon}}{\partial\boldsymbol{\phi}_l} + \ddot{\boldsymbol{\epsilon}}^{\times}\frac{\partial\boldsymbol{\epsilon}}{\partial\boldsymbol{\phi}_l} - \boldsymbol{\epsilon}^{\times}\frac{\partial\ddot{\boldsymbol{\epsilon}}}{\partial\boldsymbol{\phi}_l}\right) \tag{42}$$

K. Associating a Unit Quaternion Time Series to a Rotation Time Series

To fit a unit quaternion curve to a time series of noisy rotations can be a challenge. For the fitting procedure described in Sec. III.G, one has to first convert it to a unit quaternion time series. We will refer to this step in the initialization process as the association step.

The association of unit-length quaternions to rotations is ambiguous as Eq. (31) maps any pair of antipodal unit quaternions to the same rotation. Choosing the wrong quaternion of the pair during

association results in jumps in the unit-length quaternion curve and can cause the optimization to diverge. The obvious solution is to retrieve the minimum-length sequence of corresponding quaternions, measured as the sum of distances of all adjacent pairs.

This can be approximated by iterating over the rotation time series starting from one side and appending in each step to the target quaternion time series the one among the two unit-length quaternions equally well corresponding to the current rotation, which has minimal distance to its predecessor quaternion chosen in the step before. However, this strategy can perform badly in the presence of outliers. Our current implementation iterates through the time series of rotations choosing in each iteration the unit-length quaternion candidate that minimizes the total distance to its (up to) three predecessors. This worked well enough in our experiments, except as noted when we tested with high sensor noise (see Sec. V.B).

IV. Experiments

In this section, we apply the state representation derived previously to the spacecraft attitude estimation problem. The goal is to estimate the attitude of a reference frame attached to the vehicle body, $\underset{\rightarrow B}{\mathcal{F}}$, with respect to an inertial frame, $\underset{\rightarrow I}{\mathcal{F}}$, over a time interval of interest, $\mathcal{T} = [0, T]$. We would like to estimate a unit-length quaternion trajectory describing the rotation from the inertial frame to the body frame, $\boldsymbol{q}_{BI}(t)$, over $\mathcal{T}$.

The vehicle may have bearing sensors (we will consider one or two). A bearing sensor takes measurements at discrete time instances, $(t_m)_{m=1}^{M} \in \mathcal{T}$. The sensor's frame $\underset{\rightarrow S}{\mathcal{F}}$ is rigidly attached to the vehicle body frame. We assume that the orientation of the sensor frame with respect to the body frame, $\boldsymbol{C}_{SB}$, is known. Let $\boldsymbol{b}_I^{m\ell}$ be the known bearing of beacon ℓ (in case of sun sensors or star trackers) or the magnetic field (in case of a magnetometers) at time t_m, expressed in the inertial frame. An instantaneous measurement of this bearing (i.e., a discrete-time measurement), written $\boldsymbol{y}_{m\ell}$, is modeled as

$$\boldsymbol{y}_{m\ell} = \boldsymbol{h}(\boldsymbol{C}_{SB}\boldsymbol{C}_{\boldsymbol{q}_{BI}(t_m)}\boldsymbol{b}_I^{m\ell}) + \boldsymbol{n}_{m\ell}, \qquad \boldsymbol{n}_{m\ell} \sim \mathcal{N}(0, \boldsymbol{R}_{m\ell}) \tag{43}$$

where the bearing vector is first rotated into the sensor frame; $\boldsymbol{h}(\cdot)$ is a nonlinear observation model; $\boldsymbol{C}_q$ is the proper orthogonal matrix given by a quaternion $\boldsymbol{q}$, as in Eq. (32), here acting as the direction cosine matrix from the inertial frame to the body frame; and $\boldsymbol{n}_{m\ell}$ is zero-mean Gaussian noise with covariance $\boldsymbol{R}_{m\ell}$.

Following the standard practice of maximum likelihood estimation, we define the error term associated with this measurement to be

$$\boldsymbol{e}_{m\ell} \coloneqq \boldsymbol{y}_{m\ell} - \boldsymbol{h}(\boldsymbol{C}_{SB}\boldsymbol{C}_{\boldsymbol{q}_{BI}(t_m)}\boldsymbol{b}_I^{m\ell}) \tag{44}$$

These errors contribute to the term J_y in our overall objective function,

$$J_y \coloneqq \frac{1}{2}\sum_{m=1}^{M} \boldsymbol{e}_{m\ell}^T \boldsymbol{R}_{m\ell}^{-1} \boldsymbol{e}_{m\ell} \tag{45}$$

Later, we present simulated experiments using this common setup for the exteroceptive measurements; they differ in terms of bearing sensors and dynamics model used. Based on the latter, we distinguish two types of experiments. In the first type of experiments, we use measured angular velocities from a gyroscope. In the second, we use a dynamics model based on Euler's equation and the vehicle inertia matrix.

A. Attitude Estimation Based on Gyroscope Measurements

In experiment 1, we consider an estimation problem that neglects analytical vehicle dynamics in favor of measured dynamics from a three-axis gyroscope. For simplicity, we choose to place our vehicle body frame at the center of our gyroscope measurement frame. The gyroscope measurement model follows the one commonly used in robotics [65,66]:

$$\boldsymbol{\varpi}_g = \boldsymbol{\omega}(t_g) + \boldsymbol{b}(t_g) + \boldsymbol{n}_{g\omega}, \qquad \boldsymbol{n}_{g\omega} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_{g\omega}) \tag{46a}$$

$$\dot{\boldsymbol{b}}(t) = \boldsymbol{w}_b(t), \qquad \boldsymbol{w}_b \sim \mathcal{GP}(\boldsymbol{0}, \delta(t-t')\boldsymbol{Q}_b) \tag{46b}$$

where $\boldsymbol{\varpi}_g$ denotes the angular velocity measurement at t_g, and $\boldsymbol{\omega}$ is the angular velocity of the body as seen from the inertial frame but expressed in the body frame (see Sec. III.A for the notation). It is related to the quaternion trajectory $\boldsymbol{q}_{BI}(t)$ via Eq. (36). The gyroscope measurements are assumed to be subject to both zero-mean Gaussian measurement noise $\boldsymbol{n}_{g\omega}$ and a slowly evolving bias $\boldsymbol{b}(t)$. We will model the bias with a traditional B-spline function in $\mathbb{R}^3$ in every experiment with the same order as the attitude unit quaternion B-spline. The error for a single gyroscope measurement may be written as

$$\boldsymbol{e}_{g\omega} \coloneqq \boldsymbol{\varpi}_g - \boldsymbol{\omega}(t_g) - \boldsymbol{b}(t_g) \tag{47}$$

which becomes a term J_ω in our objective function:

$$J_\omega \coloneqq \frac{1}{2}\sum_{g=1}^{G} \boldsymbol{e}_{g\omega}^T \boldsymbol{R}_{g\omega}^{-1} \boldsymbol{e}_{g\omega} \tag{48}$$

The bias motion error may be written as

$$\boldsymbol{e}_b(t) \coloneqq \dot{\boldsymbol{b}}(t) \tag{49}$$

This error contributes to our objective function as

$$J_b \coloneqq \frac{1}{2}\int_0^T \boldsymbol{e}_b^T(t)\boldsymbol{Q}_b^{-1}\boldsymbol{e}_b(t)\,\mathrm{d}t \tag{50}$$

Together, Eqs. (45), (48), and (50) define the combined objective function $J \coloneqq J_y + J_\omega + J_b$. Let $\boldsymbol{\mathcal{Q}}$ be the stacked matrix of quaternion control vertices and $\boldsymbol{c}_b$ the stacked vector of bias spline control vertices. The control vertices for the maximum likelihood estimate of the trajectories $\boldsymbol{q}_{BI}(t)$ and $\boldsymbol{b}(t)$, $\boldsymbol{\mathcal{Q}}^\star$, $\boldsymbol{c}_b^\star$ can then approximately be found by minimizing $J(\boldsymbol{\mathcal{Q}}, \boldsymbol{c}_b)$:

$$(\boldsymbol{\mathcal{Q}}^\star, \boldsymbol{c}_b^\star) = \underset{\mathcal{Q}, c_b}{\arg\min}\, J(\boldsymbol{\mathcal{Q}}, \boldsymbol{c}_b) \tag{51}$$

B. Attitude Estimation Based on Euler's Equation

In this type of experiments, we consider a spacecraft attitude estimation problem that uses a vehicle dynamics model based on Euler's equation. The dynamics model is (expressed in the body frame)

$$\boldsymbol{I}\dot{\boldsymbol{\omega}}(t) + \boldsymbol{\omega}^\times(t)\boldsymbol{I}\boldsymbol{\omega}(t) = \boldsymbol{u}(t) + \boldsymbol{w}(t) \tag{52}$$

where t is time, $\boldsymbol{I}$ is the vehicle inertia matrix, $\boldsymbol{\omega}(t)$ is again the angular velocity of the body as seen from the inertial frame, $\boldsymbol{u}(t)$ is a known control input, and $\boldsymbol{w}$ is a zero-mean white Gaussian process with covariance $\boldsymbol{Q}_d$:

$$\boldsymbol{w} \sim \mathcal{GP}(0, \delta(t-t')\boldsymbol{Q}_d) \tag{53}$$

Following Furgale et al. [5] (Sec. III.A), we define the error for the dynamics model at time t to be

$$\boldsymbol{e}_d(t) \coloneqq \boldsymbol{I}\dot{\boldsymbol{\omega}}(t) + \boldsymbol{\omega}^\times(t)\boldsymbol{I}\boldsymbol{\omega}(t) - \boldsymbol{u}(t) \tag{54}$$

The resulting term in our objective function is

$$\boldsymbol{J}_d \coloneqq \frac{1}{2}\int_0^T \boldsymbol{e}_d^T(t)\boldsymbol{Q}_d^{-1}\boldsymbol{e}_d(t)\mathrm{d}t \tag{55}$$

Together, Eqs. (45) and (55) define the combined objective function $J \coloneqq J_y + J_d$. Let $\mathcal{Q}$ be, again, the stacked matrix of quaternion control vertices. The trajectory estimate is then given by the J-minimizing quaternion control vertices:

$$\mathcal{Q}^\star = \arg\min_{\mathcal{Q}} J(\mathcal{Q}) \tag{56}$$

C. Simulation Configurations

We simulated the attitude trajectory of a spacecraft at 350 km altitude, with a 35 deg inclination and an initial angular velocity of square norm 0.5 deg/s. This configuration is intentionally similar to those used in [4,14] so that the result can be directly compared. To stress test the dynamic model-based approach, we also simulated trajectories with a sinusoidal thruster turned on. The thruster's sinusoidal torque $\boldsymbol{\tau}$ is simulated at a time t for a given thruster factor f_τ as follows:

$$\boldsymbol{a}_\tau \coloneqq \begin{bmatrix} 0.001 & 0.0005 & 0.00075 \end{bmatrix}^T \text{ N} \cdot \text{m} \tag{57}$$

$$\boldsymbol{\omega}_\tau \coloneqq \begin{bmatrix} 0.1 & 0.05 & 0.075 \end{bmatrix}^T \frac{\text{rad}}{\text{s}} \tag{58}$$

$$\boldsymbol{\phi}_\tau \coloneqq \begin{bmatrix} 0 & \frac{\pi}{2} & -\frac{\pi}{4} \end{bmatrix}^T \text{rad} \tag{59}$$

$$\boldsymbol{\tau}_0 \coloneqq \begin{bmatrix} 0.005 & -0.005 & 0.000 \end{bmatrix}^T \text{ N} \cdot \text{m} \tag{60}$$

$$\boldsymbol{\tau}(t) \coloneqq f_\tau(\boldsymbol{a}_\tau \sin(t\boldsymbol{\omega}_\tau + \boldsymbol{\phi}_\tau) + \boldsymbol{\tau}_0) \tag{61}$$

where the sinus is applied componentwise. This torque is used as the input $\boldsymbol{u}$ in Eq. (52). The angular acceleration induced by the thruster is determined by the assumed inertia tensor of the spacecraft in the same frame:

$$\boldsymbol{I} \coloneqq \text{diag}([27 \quad 17 \quad 25]) \text{ kg} \cdot \text{m}^2 \tag{62}$$

$$\boldsymbol{\alpha}_\tau(t) \coloneqq \boldsymbol{I}^{-1}\boldsymbol{\tau}(t) \tag{63}$$

We chose our system process noise, described in Eq. (53), to be defined by $\boldsymbol{Q}_d = (1\ \mu\text{Nm})^2 \frac{1}{\text{s}}\mathbf{1}$. We simulated a three-axis magnetometer (TAM), a sun sensor, and a rate gyroscope. They were all sampled at 1 Hz, as in [4] ([14] use 0.1 Hz). We used the following parameters for the sensors.

1) Gyroscope: $\boldsymbol{R}_{g\omega} = (0.3\ \mu\frac{\text{rad}}{\text{s}})^2\mathbf{1}$, $\boldsymbol{Q}_b = (3 * 10^4\ \mu\frac{\text{rad}}{\text{s}^2})^2 \frac{1}{\text{s}}\mathbf{1}$; see Eq. (46).

2) Magnetometer: $\boldsymbol{R}_B = (50\text{ nT})^2\mathbf{1}$; see Eq. (43).

3) Sun sensor: $\boldsymbol{R}_S = (5\text{ mrad})^2\mathbf{1}$; see Eq. (43).

The simulated gyroscope bias was started at 0 or 1000 deg/h, depending on the experiment. To the best of our knowledge, these parameters are identical to [4,14], except that they did not simulate a sun sensor. For some tests, we introduced a noise factor $F \in [1, 100]$ applied to the standard deviation of the sensor measurements and the gyroscope bias process noise. For every simulation configuration, we simulated an ensemble of 100 different seeds for the pseudorandom number generators used to simulate process and sensor noise.

The typical simulated attitude trajectory is depicted in Fig. 2. It depicts the three imaginary components of the attitude quaternion $\boldsymbol{q}_{BI}$ over our maximal duration $T = 2$ h and zooms in to the first [0, 0.2] h interval as used in most of the short experiments.

D. Estimators

For our comparison, we have implemented the proposed spline estimators for continuous-time estimation capable of integrating all sensor output, including the gyroscope, as described in Sec. IV.A, and

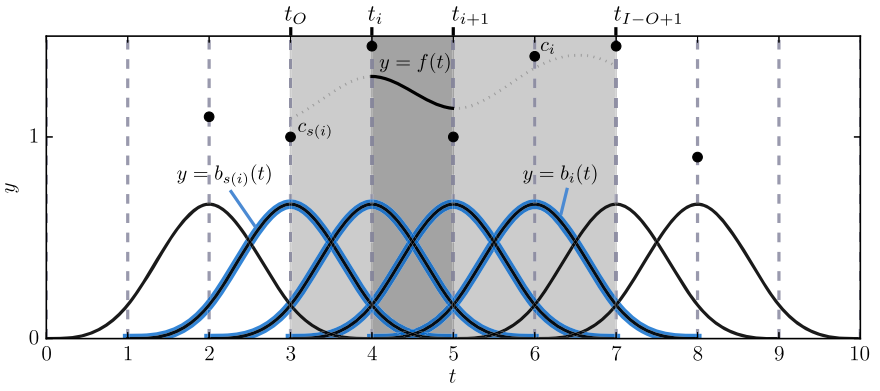


Fig. 1 Graph of a B-spline $f : [t_O, t_{I-O+1}) \to \mathbb{R}$ (with $O = 4$, $I = 11$, $t_j \coloneqq j - 1$). It shows basis functions, b_j (solid bumps), knots, t_j, (dashed lines), control vertices, c_j, (black dots). Only the highlighted basis functions contribute to f's value in the dark gray shaded area.

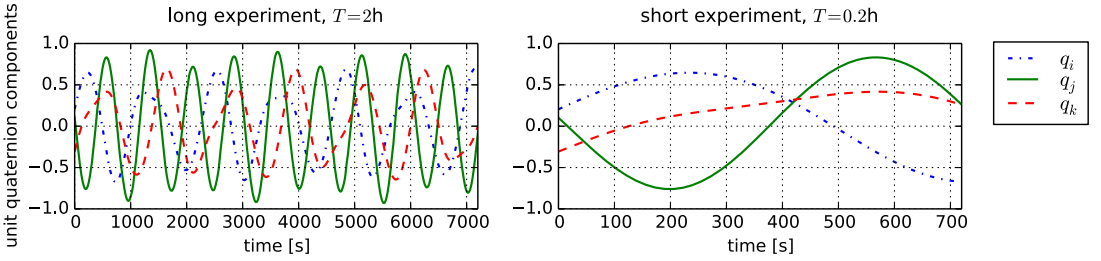


Fig. 2 Typical simulated attitude trajectory plotted as the three imaginary components of q_{BI} over the longest run time $T = 2$ h and a zoomed plot of the first 0.2 h.

the dynamics model, as described in Sec. IV.B with arbitrary spline order (starting at 2 corresponding to a polynomial order of 1, i.e., a "linear spline"). We will refer to these estimators with (label, marker) on all plots:

1) (*splXX*, *XX*): our quaternion spline implementation where XX is the B-spline order.

2) (*MEKF*, *K*): a multiplicative EKF implementation.

3) (*splVan*, *V*): a modified linear spline that uses the symplectic integration scheme described in [4] to calculate its angular velocity. Although our optimization algorithm will certainly use a different path through the parameter space of the spline, the result should be (up to the solver precision) the same as of a full implementation of the estimator described in [4].

E. Estimator Configurations

As estimator configurations we consider a specification of 1) the prior attitude distribution as normal distribution on SO(3), $\mathcal{N}(\boldsymbol{\mu}_A, \boldsymbol{P}_A)$; 2) the prior bias distribution as normal distribution on $\mathbb{R}^3$, $\mathcal{N}(\boldsymbol{\mu}_b, \boldsymbol{P}_b)$; 3) the observation duration T; 4) the sensor period, for all sensors (adjustable by subsampling the simulated sensor readings); 5) the knot spacing for the splines; and 6) the set of sensors used (rate gyro, sun, TAM).

The system dynamics model is used if and only if the gyroscope is not used.

To initialize the spline batch estimators, we start with calculating a quaternion sequence using different strategies, depending on the sensor configuration.

1) If both bearing sensors are used, we use the Q-method [6] to get a quaternion for each measurement time.

2) Otherwise, if the rate gyroscope is available, we integrate its angular velocity measurements starting at the prior's mean. To mitigate the drift of the gyroscope integration in long experiments, we initialize the full spline in subbatches of about 10 min. For each subbatch, we integrate the gyroscope, perform a full estimation with all sensor data, and then repeat the process for the next subbatch using the final value of the previous subbatch at the start of the integration time.

3) If neither is available, we use a constant sequence equal to the prior's mean.

In each case, we roughly fit the quaternion spline to this sequence to initialize the nonlinear optimization (Sec. III.G), then run a full batch optimization.

To employ the system model in the second type of experiments, we used Simpson's rule and twice as many evaluations for the numeric integration (Sec. III.H) as knots in the spline.

For the prior attitude, we chose the mean $\boldsymbol{\mu}_A$ to be 120 deg off ground truth, and the covariance matrix $\boldsymbol{P}_A = (180\ \text{deg})^2\mathbf{1}$ and $\boldsymbol{P}_b = (0.2\ \text{deg/h})^2\mathbf{1}$. The prior bias mean is always $\mathbf{0} \in \mathbb{R}^3$. We use $T = 0.2$ h as the observation time and 1 s as the default sensor period (corresponding to the 1 Hz simulation frequency). As the default knot spacing, we chose the sensor period for the linear splines (B-spline and Vandersteen) and twice the sensor period for all other splines because this gave the best results in each case. All experiment descriptions in Sec. V will be relative to this default configuration.

V. Results

We defined several experiments by picking a simulator configuration and an estimator configuration up to one open parameter. This parameter will be varied in a specific interval and make up the abscissa of the two-dimensional result plots. As ordinates, we plot mean and standard deviation over the simulated ensemble for a temporally averaged angular distance between ground truth and estimated attitude. To allow the different estimators time to converge, this temporal average is taken over the last ΔT s of the observation time T. We will denote this measure with MADE(ΔT) for mean angular distance error.

To compute the MADE(ΔT), we evaluate

$$\text{MADE}(\Delta T) \coloneqq \frac{1}{|\mathcal{U}|}\sum_{t\in\mathcal{U}} d_{\text{angle}}(\hat{\boldsymbol{q}}(t), \boldsymbol{q}(t)) \tag{64}$$

where d_{angle} denotes the angular distance (in degrees) of two unit quaternions (≔ Riemannian distance of the represented rotations); $\hat{\boldsymbol{q}}(t)$ is the estimated attitude quaternion at time t; $\boldsymbol{q}(t)$ is the simulated attitude quaternion; $\mathcal{U} \subset]T-\Delta T, T]$ is the set of all time instances for which both $\hat{\boldsymbol{q}}(t)$ and $\boldsymbol{q}(t)$ are available; and $|\mathcal{U}|$ is its cardinality. For all continuous estimators, $\mathcal{U}$ are the time instances in $]T-\Delta T, T]$ for which $\boldsymbol{q}(t)$ was simulated (at 10 Hz). The discrete MEKF always runs on a regular subset (depending on the experiment) of the simulated times and hence defines the $\mathcal{U}$ for its MADE evaluation.

This approximates a temporally averaged angular distance because $\mathcal{U}$ is always evenly spaced, which yields for two continuous unit quaternion-valued functions $\boldsymbol{q}_1$, $\boldsymbol{q}_2$ on $[T-\Delta T, T]$

$$\frac{1}{|\mathcal{U}|}\sum_{t\in\mathcal{U}} d_{\text{angle}}(\boldsymbol{q}_1(t), \boldsymbol{q}_2(t)) \simeq \frac{1}{\Delta T}\int_{T-\Delta T}^{T} d_{\text{angle}}(\boldsymbol{q}_1(t), \boldsymbol{q}_2(t))\,\mathrm{d}t \tag{65}$$

We decided to use the Riemannian distance (Eluclidean norm of the relative rotation vector) instead of the root sum squared (RSS) of roll, pitch, and yaw used in Vandersteen et al. [4] and Crassidis and Markley [14] because the latter is not isotropic and would render the result dependent of the current pose of the spacecraft. For small errors, they are very similar in magnitude, and thus the results are still comparable.

Except for this difference in the error metric at a point in time, MADE(T) is like the integral cost J defined in Eq. (46) of Crassidis and Markley [14].

A. Gyroscope Based

First, the experiments based on gyroscope measurements, corresponding to Sec. IV.A.

Figure 3 shows the results for the extreme initial bias test case of [4]. This experiment uses an initial bias of 1000 deg/h and only a gyroscope and magnetometer (no sun sensor). It shows how the MEKF recovers very slowly from the bad prior (120 deg away from ground truth), whereas all spline estimators have good results after 0.5 h, at the latest. The fourth-order spline has more problems with the high bias error than the linear ones and the sixth-order. The former is probably mainly an effect of the fact that the linear splines have twice as many knots. Qualitatively, these results fit the results of [4] as far as they overlap. However, there are two major differences. First, in our experiment, the MEKF was able to converge for all random seeds, just very slowly. And second, the limiting accuracy was about 5×10^{-3} deg in our case and 10^{-3} deg in theirs. These differences are most likely caused by differences in the simulated sensor data

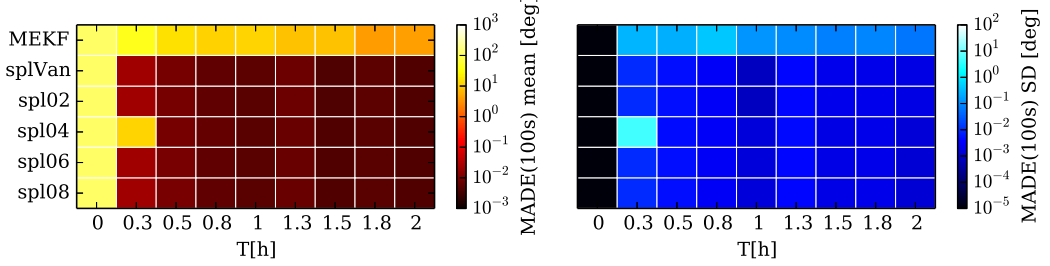


Fig. 3 MADE(100 s) over the observation time T given a simulated initial bias of 1000 deg/h. $T \in$ [0,2] h. Sensors: rate gyroscope, magnetometer.

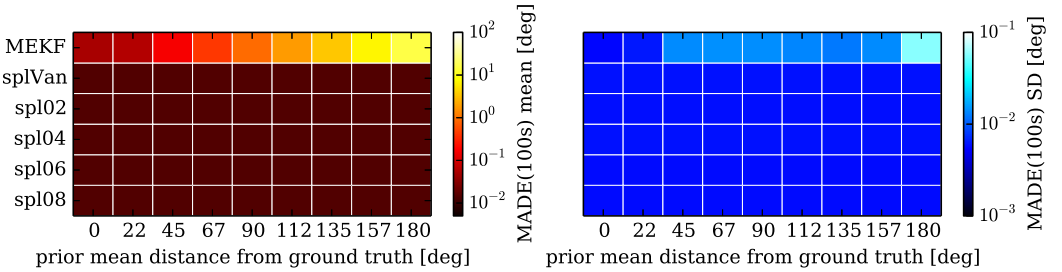


Fig. 4 MADE(100 s) over the distance of the prior mean to ground truth. $T = 0.2$ h. Sensors: rate gyroscope, magnetometer.

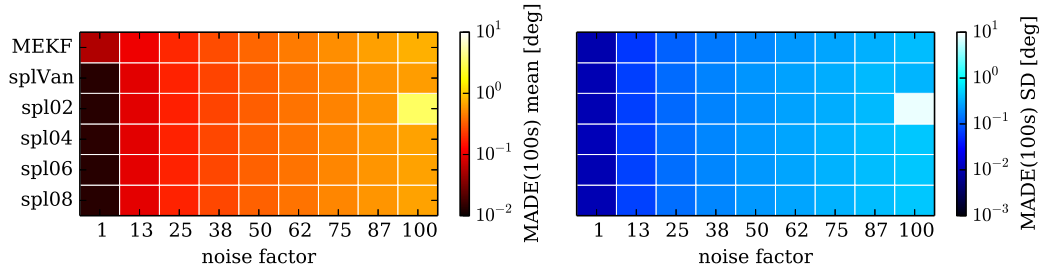


Fig. 5 MADE(100 s) over a factor multiplying the standard deviation of all sensor noise. $T = 0.2$ h. Sensors: rate gyroscope, sun sensor, magnetometer.

because we do not know the initial angular velocity or the simulated system noise used in [4].

In Fig. 4, we show the response of each estimator to a bad prior mean after $T = 0.2$ h of simulation time. Here, all batch estimators converge with essentially the same accuracy, whereas the Kalman filter has trouble converging as the prior mean failure gets large. The low variance of the MEKF measurements indicates that the MEKF's estimate is not only spoiled by noise but has systematic problems when facing a prior with a mean far off the truth.

In Fig. 5, we test the effect of sensor noise on each estimator. The splines excel in low noise range but are on par with the MEKF when sensor noise increases. The MEKF converges in every case due to the high sensor rate.

In Fig. 6, we tested the effect on higher angular velocity and acceleration due to activated thruster. Only the Kalman filter and the Vandersteen spline are noticeably affected by this addition. In the case of the Vandersteen spline, we suspect that this is due to the use of the symplectic integration scheme to approximate the angular velocity. In contrast, state derivatives are exact and analytical for all B-spline solutions.

In Fig. 7, we examined the effect of the sensor rate (defined by its period). Again, the Kalman filter and the Vandersteen spline are much more negatively affected. The similarity between Figs. 6 and 7 (especially in the beginning) is because the accuracy of the simulated sensor is not affected by the spacecraft's velocity; increasing the sensor period has virtually the same effect as speeding up the whole motion and leaving the period fixed. The effect of the thruster at 100% power roughly corresponds to the sensor period of 20 s. The fourth- and sixth-order splines become worse than the linear spline (spline order 2) because they have only half the number of knots, and in this experiment they all reach their bounds of their expressiveness (the frequency of motion they can represent) because they have to model a much longer trajectory with a fixed number of knots.

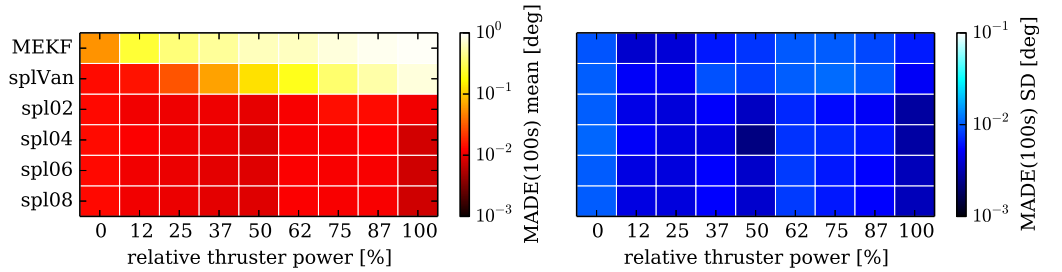


Fig. 6 MADE(100 s) over the thruster's power factor in percentage. $T = 0.2$ h. Sensors: rate gyroscope, sun sensor, magnetometer.

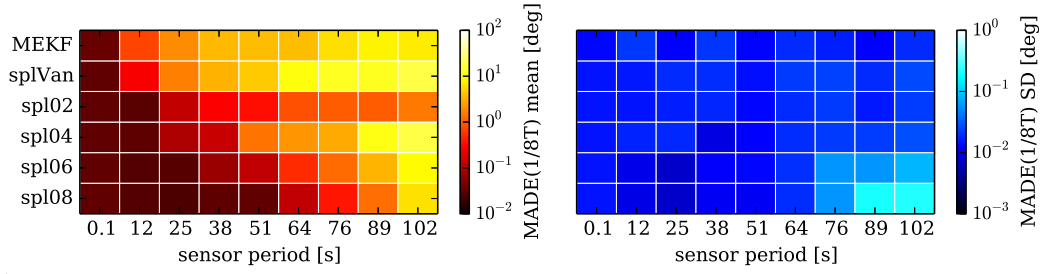


Fig. 7 MADE($\frac{1}{8}T$) over the sensors' period. T is determined by a fixed amount of 64 measurements per sensor. Sensors: rate gyroscope, sun sensor, magnetometer.

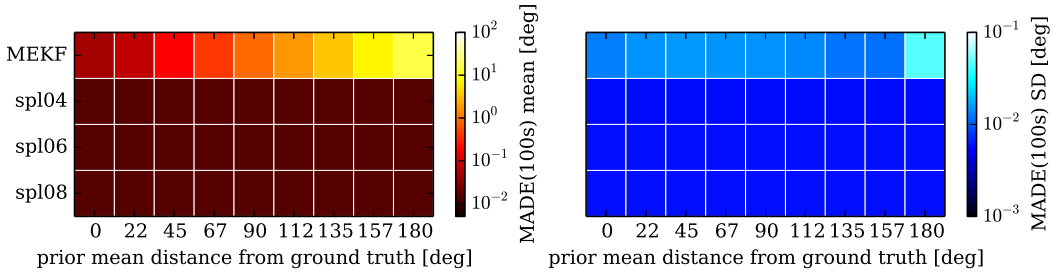


Fig. 8 MADE(100 s) over the angular distance error of the prior. $T = 0.2$ h. Sensors: magnetometer.

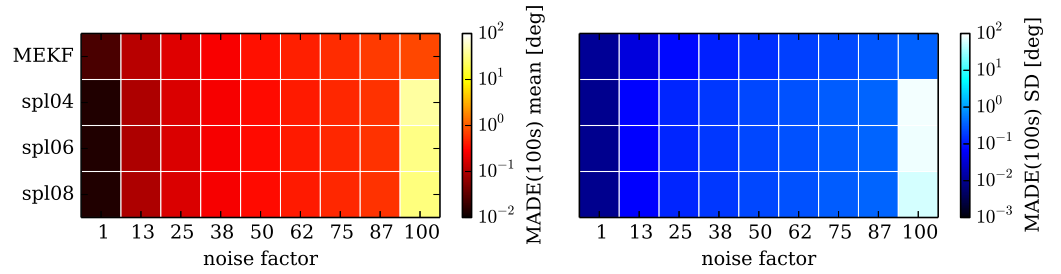


Fig. 9 MADE(100 s) over a factor multiplying the standard deviation of all sensor noise. $T = 0.2$ h. Sensors: sun sensor, magnetometer.

B. Euler's Equation Based

Next, we present the experiments based on Euler's equation. Here, we cannot use the two linear spline versions we used in former section because their angular acceleration is zero or undefined, which does not allow a straightforward application of the system model. In this section, we skip the first convergence experiment (corresponding to Fig. 3) because we do not have a initialization method reliable enough for the extreme initial bias case. Apart from that, we first follow roughly the same program as for the gyroscope-based experiments.

In Fig. 8, we show the response of each estimator to a bad prior mean after $T = 0.2$ h of simulation time. It shows again how all splines perform superior to the MEFK.

In Fig. 9, we test the effect of sensor noise on each estimator. The system-model-based spline estimators excel in the low noise range. Their bad performance for the highest noise case is caused by wrongly signed quaternions in the result of the quaternion association step (Sec. III.K). In the case of such a bad initialization, the system model pulls strongly toward an incorrect solution. When we use gyroscope measurements instead of the system model (Fig. 5), the estimator is able to find its way back to the correct solution.

In Fig. 10, we compare the performance of the estimators while varying the intensity of the thruster. Only the Kalman filter and the fourth-order spline are noticeably affected. Here, the high-order splines clearly excel. The poor performance of the fourth-order spline is caused by its bad compatibility with the ordinary differential equation (ODE) given by the system model, especially when the thruster is activated. This incompatibility pulls the estimator away from the good solutions because the low noise on the dynamics makes the estimator trust the system model more than the sensors. When we increase the noise $\boldsymbol{w}$ in the simulated system [Eq. (52)], this difference continuously decreases. We will analyze that effect further in Sec. V.C.

In Fig. 11, we tested the effect of the sensor rate (as in Fig. 7). Again, the Kalman filter is much more negatively affected than the high-order splines. Figure 10 is again quite similar to the first 20 s. The bad effect on the fourth-order spline is more severe than in the gyroscope experiment. We believe that this is again caused by the incompatibility between cubic B-splines and the system model. This difference becomes small when we increase the system noise.

C. Discussion

1. *Impact of the B-Spline Order*

The most surprising discovery we had while evaluating these experiments was the strong influence of the spline order on the estimation results. The numbers of parameters for a B-spline with N usable segments (between the knots t_O and t_{O+N}) and spline order O is proportional to $O + N$. These parameters N and O, are the tuning parameters through which one may influence the expressiveness of a B-spline curve, i.e., which trajectories can be expressed. This does not only affect how close the splines can approximate the true trajectory (see Sec. V.C.5) but also how much a continuous-time batch estimator is supported in finding realistic solutions given the measurements. For a given system (represented mathematically by a stochastic ODE), the leap from cubic to quintic piece wise polynomials (requiring only two more control vertices for any curve length) may result in a much better fit to the probable motions produced by the system (solutions of the ODE) than adding two more segments. B-splines (with increasing knots) are C^{O-1} continuous at the knots and infinitely differentiable everywhere else. This means that increasing the number of knots, N, creates more times at which

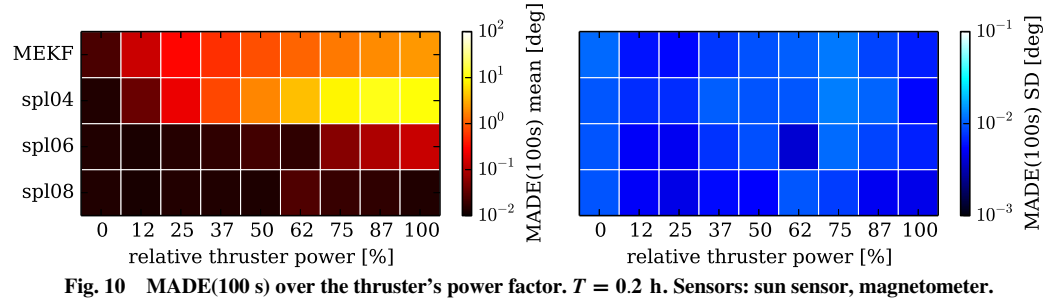


Fig. 10 MADE(100 s) over the thruster's power factor. $T = 0.2$ h. Sensors: sun sensor, magnetometer.

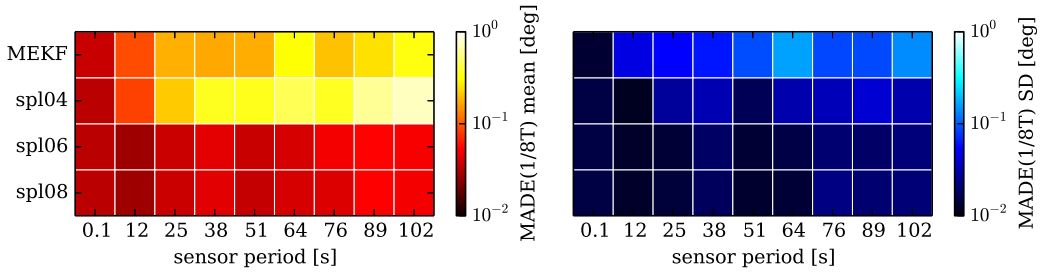


Fig. 11 MADE($\frac{1}{8}T$) over the sensors' period. T is determined by a fixed amount of 64 measurements per sensor. Sensors: sun sensor, magnetometer.

the curve is not infinitely differentiable, whereas leaving N fixed and increasing the spline order O increases how often the curves are differentiable everywhere. For our spacecraft dynamics, the fourth-order spline produces significantly worse estimates than a sixth-order spline.

To demonstrate the role of spline order vs segment number for our batch estimator, we came up with the following experiment. In Fig. 12, we assessed the estimation accuracy while changing the uniform knot spacing Δt_{knots} for different spline orders up to 13. This implies dividing the run time $T = 0.2$ h into different numbers of segments, $N = \text{ceil}(720s/\Delta t_{\text{knots}})$, ranging from 115 to 8. The higher-order splines ($\geq$12) were more difficult to initialize for high knot spacing (above 80 s), as mentioned in Sec. III.G, and required to raise the acceleration penalty. The drastic effect of the order catches the eye. In the system model experiment, increasing the order from 4 to 5 (adding one quaternion more to the parameters) corresponds roughly to reducing the number of segments from 20 to 10, yielding a reduction of parameters from 24 to 15 quaternions. In the gyroscope experiment, the effect is lower but still significant; the step from 4 to 5 corresponds to a decrease in segments from 20 to 12.

In the high-accuracy range, the result is even more extreme. For example, in the gyroscope-based experiment, a fourth-order spline requires at least eight times more segments to achieve the same accuracy as a 12th-order spline. In the system model experiment, it is 16 times more.

In these plots, the linear splines seem even worse than in the other experiments because they are using the same knot spacing as the other splines. Recall that, in the other experiments, they used twice as many knots.

2. *Computational Cost*

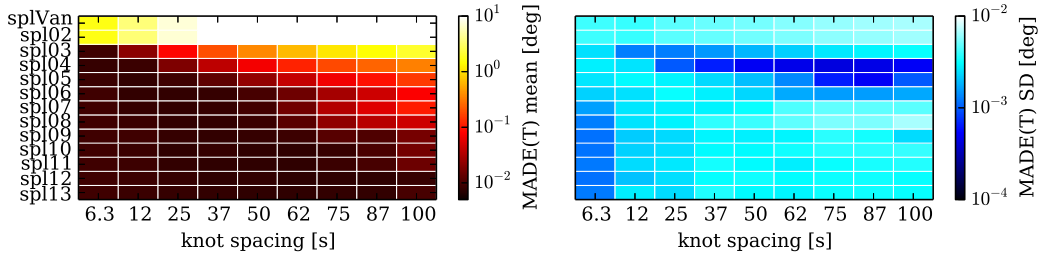
Of course, the estimation accuracy needs to be compared with the corresponding computational cost. Because the Jacobian evaluation has a big impact, it is hard to theoretically derive the computational complexity. Thus, we only provide single-core CPU time measurements on a workstation in Fig. 13. It shows surprisingly little effect of the number of segments on a single iteration. The majority of time is spent on the evaluation of the Jacobians, which only depends on the number of measurements (fixed here) and spline order O.

3. *Convergence Behavior*

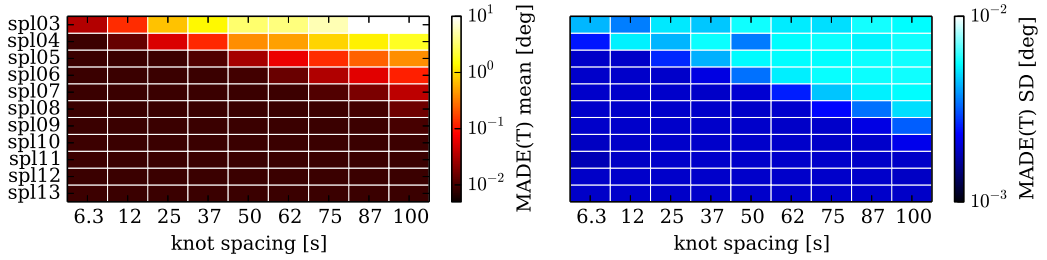
In most of the experiments, the convergence behavior is as follows. There is typically a big step after the first iteration, followed by minor improvements in subsequent iterations, except for the linear splines that converge slower and mutually very similar. That is why in Fig. 14 we show the convergence behavior in a very difficult setup instead. The B-spline gets initialized with a constant value, which is 120 deg off the simulated initial attitude, and the estimators are only using the magnetometer and the rate gyroscope. In the gyroscope case, the order seems to heavily influence if and how long it takes to recover from the bad initial value but without any obvious rule how. In the system model experiment, the typical independence of the order is clearly visible, except for very high-order B-splines (here 10th order). The sensor rates almost have no impact on the convergence behavior. However, the assumed sensor noise has a surprisingly big impact. Interestingly, increasing the sensor noise in the gyroscope case will first increase the convergence speed, even in absolute accuracy. Further rising the noise magnitude results in slower convergence, and eventually, the convergence behavior will become independent of the order.

4. *Under-Determined Case*

To demonstrate how well the system model can prevent the splines from overfitting, we run another experiment, very similar to Fig. 12, by starting the knot spacing below the sensor period (1 s) and

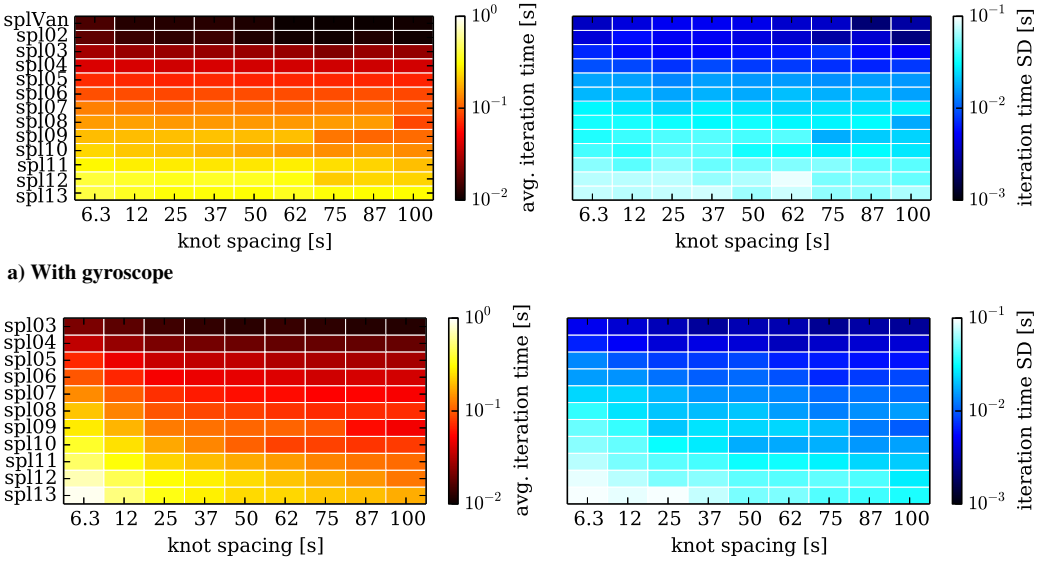


a) With gyroscope



b) With system model

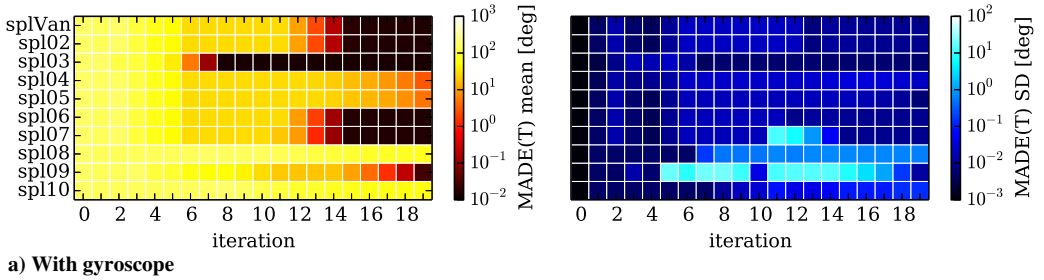
Fig. 12 MADE(T) over the time between knots. $T = 0.2$ h. Sensors: sun sensor, magnetometer.



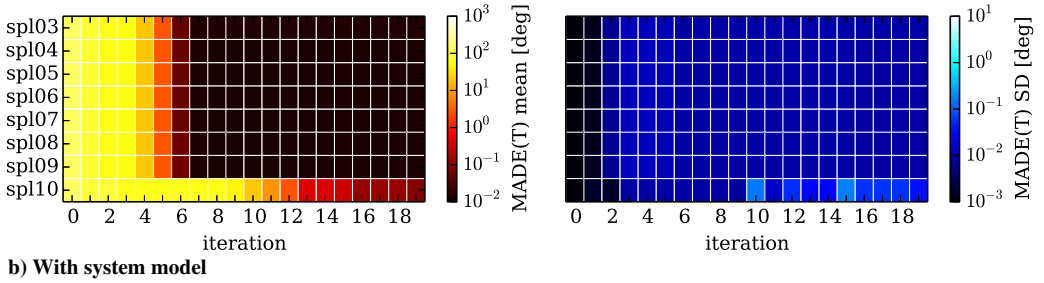
a) With gyroscope

b) With system model

Fig. 13 Time for one iteration over knot spacing on a single Intel Xeon E5 core at 3.6 GHz. $T = 0.2$ h, yielding 720 readings per sensor. Sensors: sun sensor, magnetometer.

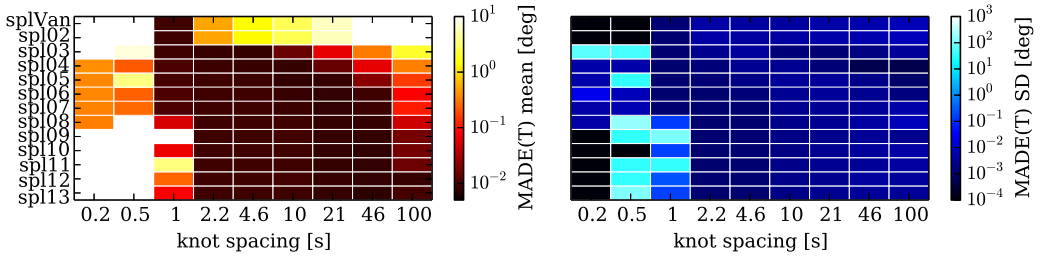


a) With gyroscope

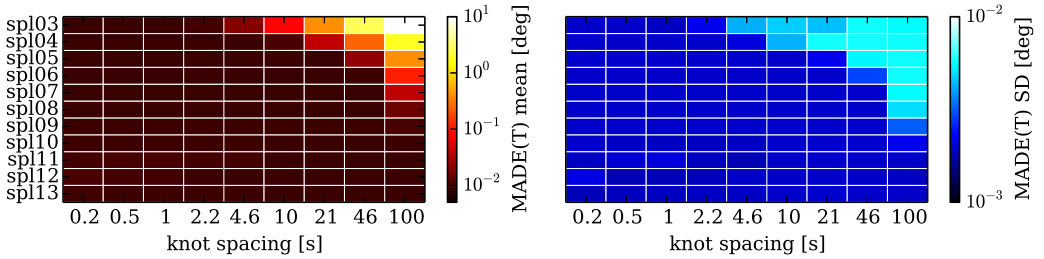


b) With system model

Fig. 14 MADE(T) over iterations (0 = after initialization) after a very bad B-spline initialization: constant, at 120 deg off the initial attitude. $T = 0.2$ h. Sensors: rate gyroscope, magnetometer.



a) With gyroscope



b) With system model

Fig. 15 MADE(T) over the time between knots on a logarithmic scale. The sensor rate is at $1s = 10^0$ s. $T = 0.2$ h. Sensors: sun sensor, magnetometer.

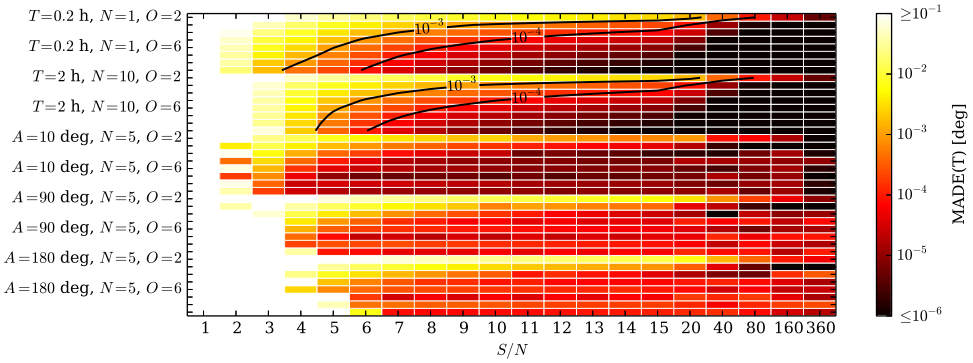


Fig. 16 Artificial sinusoidal attitude trajectory, turning around a fixed axis with amplitude $A \in \{10,45,90\}$ deg, and period of 1 s is approximated with evenly spaced until-length quaternion B-splines of various order $O \in \{2 \ldots 9\}$ and number of segments S. The abscissa is the fraction S/N, with $N = 5$ the number of simulated periods.

exponentially increasing the number of knots. The result is depicted in Fig. 15. Obviously, the gyroscope-based estimation becomes inaccurate and unstable when the knot rate is below or close to the sensor rate (10^0 s) while the system model based estimation keeps maximal accuracy.

5. Approximation Capacity Analysis

In our estimation experiments, we do not distinguish between estimation error and approximation error. In a next step, we therefore determine the smallest possible analyze the pure approximation error in a separate experiment.

Figure 16 shows the result of this experiment. The color encoded approximation error is clamped to the interval $[10^{-1}, 10^{-6}]$ deg for better readability.

We analyze two types of trajectories 1) the simulated space craft trajectories as used for the estimation experiments ($T \in \{0.2, 2\}$ h) and 2) sinusoidal attitude trajectories, turning around a fixed axis with amplitute, $A \in \{10, 90, 180\}$ deg and $N = 5$ periods of 1 s ($T = 5$ s). The second type we include for comparison and to give some intuition about how to approximate accelerated trajectories.

Each trajectory is approximated with evenly spaced unit-length quaternion B-splines of varying order, $O \in \{2 \ldots 9\}$, and varying number of valid segments $S := I - 2O + 1$. When increasing S, typically there is a big leap to good accuracy between $S = 3N$and $S = 6N$.

All our estimation experiments have $S/N \geq 360$ ($\simeq 2$ s knot spacing) except for the experiments in Figs. 12 and 15, which have variable knot spacing. The approximation accuracy $S/N = 360$, is clearly better than 10^{-4} deg. It follows that for these experiments the approximation error is negligible, because their accuracy is always even worse than 10^{-3} deg. The same is true for all the other estimation experiments with higher knot spacing but to show this requires a detailed sub-experiment wise comparison skipped here for the sake of brevity.

VI. Conclusions

In this paper, an existing unit-length quaternion B-spline formulation is extended to any order and all Lie groups, and all necessary Jacobian formulas needed for batch continuous-time state estimation on Lie groups are derived both in the general case and in the specific case for attitude estimation using unit-length quaternion splines, a singularity-free continuous-time attitude representation. The performance of these curves is evaluated for estimating spacecraft attitude against a state-of-the-art approach. B-splines are shown to have equal or superior performance over all test cases and provide two key tuning parameters, the number of knots and the spline order, that an engineer can use to trade off accuracy and computational efficiency when choosing a spline representation for a given estimation problem.

Appendix A: Proofs for Angular Velocity Equations

Let $\boldsymbol{r}(t, \boldsymbol{v}) := \mathrm{Rot}_{\boldsymbol{p}(t)}(\boldsymbol{v})$ [see Eq. (31)] describe the rotation by the unit-length quaternion $\boldsymbol{p}(t)$ of any coordinate triple $\boldsymbol{v} \in \mathbb{R}^3$ expressed in the frame $\underset{\rightarrow a}{\mathcal{F}}$. Then, the following equation defines the angular velocity $\boldsymbol{\omega}_a(t)$ in the same frame $\underset{\rightarrow a}{\mathcal{F}}$ for this rotation of the coordinates $\boldsymbol{v}$:

$$\frac{\partial \boldsymbol{r}(t, \boldsymbol{v})}{\partial t} = \boldsymbol{\omega}_a(t)^{\times} \boldsymbol{r}(t, \boldsymbol{v})$$

Please note that we introduced $\boldsymbol{p}$ to represent the rotation actively in $\underset{\rightarrow a}{\mathcal{F}}$ as opposed to the definition of $\boldsymbol{q}_{ba}$, which passively describes the change of coordinates from $\underset{\rightarrow a}{\mathcal{F}}$ to $\underset{\rightarrow b}{\mathcal{F}}$. We start this way because the physical notion of angular velocity is conceptually closer to the active rotation concept. With $\tilde{\boldsymbol{v}} := \boldsymbol{V}\boldsymbol{v}$, we have

$$\frac{\partial \boldsymbol{r}(t, \boldsymbol{v})}{\partial t} = \frac{\partial}{\partial t} \boldsymbol{W}(\boldsymbol{p}(t) \tilde{\boldsymbol{v}} \boldsymbol{p}^{-1}(t)) = \boldsymbol{W}(\dot{\boldsymbol{p}}(t) \tilde{\boldsymbol{v}} \boldsymbol{p}^{-1}(t) + \boldsymbol{p}(t) \tilde{\boldsymbol{v}} \dot{\boldsymbol{p}}^{-1}(t)) \tag{A1}$$

$$= \boldsymbol{W}(\dot{\boldsymbol{p}}(t) \boldsymbol{p}^{-1}(t) \boldsymbol{p}(t) \tilde{\boldsymbol{v}} \boldsymbol{p}^{-1}(t) - \boldsymbol{p}(t) \tilde{\boldsymbol{v}} \boldsymbol{p}^{-1}(t) \dot{\boldsymbol{p}}(t) \boldsymbol{p}^{-1}(t)) \tag{A2}$$

$$= \boldsymbol{W}(\dot{\boldsymbol{p}}(t) \boldsymbol{p}^{-1}(t) \boldsymbol{V}\boldsymbol{r}(t, \boldsymbol{v})) + \underbrace{\boldsymbol{W}(-\boldsymbol{r}(t, \boldsymbol{v}) \dot{\boldsymbol{p}}(t) \boldsymbol{p}^{-1}(t))}_{= \boldsymbol{W}(\dot{\boldsymbol{p}}(t) p^{-1}(t) r(t, v))} \tag{A3}$$

$$= 2\boldsymbol{W}(\dot{\boldsymbol{p}}(t) \boldsymbol{p}^{-1}(t) \boldsymbol{V}\boldsymbol{r}(t, \boldsymbol{v})) \tag{A4}$$

$$= 2\boldsymbol{W}(\dot{\boldsymbol{p}}(t) \boldsymbol{p}^{-1}(t))^{\times} \boldsymbol{r}(t, \boldsymbol{v}) \tag{A5}$$

Because the latter equation has to be true for any $\boldsymbol{v}$, it follows that (omitting the time parameter)

$$\boldsymbol{\omega}_a = 2\boldsymbol{W}(\dot{\boldsymbol{p}} \boldsymbol{p}^{-1}) \tag{A6}$$

In the case that $\underset{\rightarrow b}{\mathcal{F}}$ is $\underset{\rightarrow a}{\mathcal{F}}$ rotated by $\boldsymbol{p}$, it holds that $\boldsymbol{q}_{ba} = \pm \boldsymbol{p}^{-1}$. Inserting in Eq. (A6) yields, when omitting the subscript of $\boldsymbol{q}_{ba}$,

$$\boldsymbol{\omega}_a = 2\boldsymbol{W}(\dot{\boldsymbol{p}} \boldsymbol{p}^{-1}) = 2\boldsymbol{W}(\dot{\boldsymbol{q}}^{-1} \boldsymbol{q}) = -2\boldsymbol{W}(\boldsymbol{q}^{-1} \dot{\boldsymbol{q}}) \tag{A7}$$

because $\dot{\boldsymbol{q}}^{-1} = -\boldsymbol{q}^{-1} \dot{\boldsymbol{q}} \boldsymbol{q}^{-1}$. This proves Eq. (35).

Angular velocity expressed in $\underset{\rightarrow b}{\mathcal{F}}$ can be acquired by transforming the coordinates using $\boldsymbol{q}_{ba}$ by applying Eq. (34):

$$\boldsymbol{\omega}_b = \boldsymbol{W}(\boldsymbol{q}(\boldsymbol{V}\boldsymbol{\omega}_a)\boldsymbol{q}^{-1}) = -2\boldsymbol{W}(\boldsymbol{q}(\boldsymbol{V}\boldsymbol{W}(\boldsymbol{q}^{-1}\dot{\boldsymbol{q}}))\boldsymbol{q}^{-1}) = -2\boldsymbol{W}(\dot{\boldsymbol{q}}\boldsymbol{q}^{-1}) \tag{A8}$$

because $\boldsymbol{V}\boldsymbol{W}$ acts as identity on the pure imaginary value $\boldsymbol{q}^{-1}\dot{\boldsymbol{q}}$. It must be pure imaginary because complex conjugation (denoted with a hat in the following) negates it because $\boldsymbol{q}$ is of unit length ($\Rightarrow \boldsymbol{q}^{-1} = \hat{\boldsymbol{q}}$):

$$\widehat{\hat{\boldsymbol{q}}\,\dot{\boldsymbol{q}}} + \hat{\boldsymbol{q}}\,\dot{\boldsymbol{q}} = \hat{\dot{\boldsymbol{q}}}\boldsymbol{q} + \hat{\boldsymbol{q}}\,\dot{\boldsymbol{q}} = \dot{\hat{\boldsymbol{q}}}\boldsymbol{q} + \hat{\boldsymbol{q}}\,\dot{\boldsymbol{q}} = \frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{q}}\boldsymbol{q} = 0$$

The second equivalence follows from the fact that taking derivative commutes with complex conjugation. And Eq. (A8) proves Eq. (36).

Appendix B: Proofs for Unit-Length Quaternion Exponential and Logarithm Maps' Jacobian Formulas

Let $\boldsymbol{\phi} \in \mathbb{R}^3 \setminus \{\boldsymbol{0}\}$ represent a purely imaginary quaternion and $\phi \coloneqq \|\boldsymbol{\phi}\|$, $\phi\boldsymbol{a} \coloneqq \boldsymbol{\phi}$. First, we derive the exponential map's Jacobian $\boldsymbol{S}(\boldsymbol{\phi})$, as given in Eq. (39) by starting with Eq. (21):

$$\boldsymbol{S}(\boldsymbol{\phi}) = \sum_{k=0}^{\infty} \frac{(-1)^k}{(k+1)!}(\mathrm{ad}_{\boldsymbol{\phi}})^k \tag{B1a}$$

$$= (\mathrm{ad}_{\boldsymbol{\phi}})^0 + \sum_{k=1}^{\infty} \frac{(-1)^{2k-1}}{(2k-1+1)!}(\mathrm{ad}_{f\boldsymbol{\phi}})^{2k-1} + \sum_{k=0}^{\infty} \frac{(-1)^{2k+2}}{(2k+2+1)!}(\mathrm{ad}_{\boldsymbol{\phi}})^{2k+2} \tag{B1b}$$

$$= \boldsymbol{1} + \sum_{k=1}^{\infty} \frac{-1}{(2k)!}((-1)^{k-1}2^{2k-1}\phi^{2k-1}\boldsymbol{a}^{\times}) + \sum_{k=0}^{\infty} \frac{1}{(2k+2+1)!}((-1)^k(2\phi)^{2k+2}\boldsymbol{a}^{\times}\boldsymbol{a}^{\times}) \tag{B1c}$$

$$= \boldsymbol{1} - \frac{1}{\phi}\sin^2\phi\boldsymbol{a}^{\times} + \frac{-1}{2\phi}\left(\sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)!}(2\phi)^{2k+1} - 2\phi\right)\boldsymbol{a}^{\times}\boldsymbol{a}^{\times} \tag{B1d}$$

$$= \boldsymbol{1} - \frac{1}{\phi}\sin^2\phi\boldsymbol{a}^{\times} + \left(1 - \frac{1}{2\phi}\sin(2\phi)\right)\boldsymbol{a}^{\times}\boldsymbol{a}^{\times} \tag{B1e}$$

We use to get Eq. (B1c) that for all $\boldsymbol{w} \in \mathbb{H}$, also purely imaginary, $\mathrm{ad}_{\boldsymbol{\phi}}\boldsymbol{w} = [\boldsymbol{\phi}, \boldsymbol{w}] = \boldsymbol{\phi}\boldsymbol{w} - \boldsymbol{\phi}\boldsymbol{w} = 2\boldsymbol{\phi} \times \boldsymbol{w}$, which implies that $2\boldsymbol{\phi}^{\times}$ is the matrix representation of $\boldsymbol{\phi}$'s adjoint action in $\mathcal{B}_{\mathfrak{g}} = (\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k})$, and that for the cross product it holds $(\boldsymbol{a}^{\times})^3 = -\boldsymbol{a}^{\times}$, which implies $(\boldsymbol{\phi}^{\times})^{2k+1} = (-1)^k\phi^{2k+1}\boldsymbol{a}^{\times}$ and $(\boldsymbol{\phi}^{\times})^{2k+2} = (-1)^k\phi^{2k+2}\boldsymbol{a}^{\times}\boldsymbol{a}^{\times}$ for $k \in \mathbb{N}$. To get Eqs. (B1d) and (B1e), we only use well known power series expansions for $\sin^2$ and $\sin$, respectively.

Next, we prove Eq. (40) by showing that $\boldsymbol{S}(\log(\boldsymbol{q}))\boldsymbol{L}(\boldsymbol{q}) = \boldsymbol{1}$ for $\boldsymbol{q} \neq \pm j$. Let $\boldsymbol{\phi} \coloneqq \log(\boldsymbol{q})$, then

$$\boldsymbol{S}(\boldsymbol{\phi})\boldsymbol{L}(\boldsymbol{q}) = \left(\boldsymbol{1} - \frac{1}{\phi}\sin^2\phi\boldsymbol{a}^{\times} + \left(1 - \frac{1}{2\phi}\sin(2\phi)\right)\boldsymbol{a}^{\times}\boldsymbol{a}^{\times}\right) \times \left(\boldsymbol{1} + \boldsymbol{\phi}^{\times} + \left(1 - \frac{\phi}{\tan\phi}\right)\boldsymbol{a}^{\times}\boldsymbol{a}^{\times}\right) \tag{B2a}$$

$$= \boldsymbol{1} + \left(\phi - \frac{1}{\phi}\sin^2\phi + \frac{1}{\phi}\sin^2\phi\left(1 - \frac{\phi}{\tan\phi}\right) - \phi\left(1 - \frac{1}{2\phi}\sin(2\phi)\right)\right)\boldsymbol{a}^{\times} \tag{B2b}$$

$$+ \left(1 - \frac{\phi}{\tan\phi} - \sin^2\phi + 1 - \frac{1}{2\phi}\sin(2\phi) - \left(1 - \frac{1}{2\phi}\sin(2\phi)\right)\left(1 - \frac{\phi}{\tan\phi}\right)\right)\boldsymbol{a}^{\times}\boldsymbol{a}^{\times} \tag{B2c}$$

$$= \boldsymbol{1} + \left(-\frac{\sin^2\phi}{\tan\phi} + \frac{1}{2}\sin(2\phi)\right)\boldsymbol{a}^{\times} + \left(1 - \sin^2\phi - \frac{1}{2}\sin(2\phi)\frac{1}{\tan\phi}\right)\boldsymbol{a}^{\times}\boldsymbol{a}^{\times} \tag{B2d}$$

$$= \boldsymbol{1} \tag{B2e}$$

In the first step, only $(\boldsymbol{a}^{\times})^3 = -\boldsymbol{a}^{\times}$ is used. In the last step, we used $\frac{1}{2}\sin(2\phi) = \cos\phi\,\sin\phi$, $\tan = \sin/\cos$, and $1 = \sin^2 + \cos^2$.

Appendix C: Proof of Equation (22)

$$\begin{aligned}\frac{\partial}{\partial\boldsymbol{\phi}_l}\Phi_{\bar{d}}^{-1}(\boldsymbol{d}) &= \frac{\partial}{\partial\boldsymbol{\phi}_l}\log(\boldsymbol{d}\bar{\boldsymbol{d}}^{-1}) = \boldsymbol{W}\frac{\partial}{\partial\boldsymbol{\phi}_l}(\boldsymbol{d}\bar{\boldsymbol{d}}^{-1}) \\ &= \boldsymbol{W}\frac{\partial}{\partial\boldsymbol{\phi}_l}(\bar{\boldsymbol{g}}_{k-1}^{-1}\exp(-\boldsymbol{\phi}_{k-1})\exp(\boldsymbol{\phi}_k)\bar{\boldsymbol{g}}_k(\bar{\boldsymbol{g}}_{k-1}^{-1}\bar{\boldsymbol{g}}_k)^{-1}) \\ &= \boldsymbol{W}(\bar{\boldsymbol{g}}_{k-1}^{-1})^L(\bar{\boldsymbol{g}}_{k-1})^R\frac{\partial}{\partial\boldsymbol{\phi}_l}\exp(-\boldsymbol{\phi}_{k-1})\exp(\boldsymbol{\phi}_k) \\ &= \underbrace{\boldsymbol{W}(\bar{\boldsymbol{g}}_{k-1}^{-1})^L(\bar{\boldsymbol{g}}_{k-1})^R\boldsymbol{V}}_{=C(\bar{\boldsymbol{g}}_{k-1}^{-1})}(\delta_{l,k} - \delta_{l,k-1})\boldsymbol{1}\end{aligned}$$

Appendix D: Proof of Theorem 1

Theorem 1: $\mathcal{A}^L$ is a sub-$\mathbb{R}$-algebra of $\mathbb{R}^{\dim\mathcal{A}\times\dim\mathcal{A}}$; $\mathcal{G}^L$ is a matrix Lie group embedded in $\mathcal{A}^L$; and this pair is fully isomorphic to the pair of $\mathcal{G}$ and $\mathcal{A}$.

Proof: It is enough to show that $\cdot^L$ is an algebra-monomorphism into $\mathbb{R}^{\dim\mathcal{A}\times\dim\mathcal{A}}$ because its image $\mathcal{A}^L$ is then a subalgebra and the mapping an algebra-isomorphism onto it. The existence of this isomorphism yields already that $\mathcal{A}$ is isomorphic to $\mathcal{A}^L$, that $\mathcal{G}^L$ is a multiplicative subgroup in it, and its restriction to $\mathcal{G}$, $\cdot^L|_{\mathcal{G}}$, must be a group isomorphism. Because any vector space isomorphism is also smooth with respect to the canonical differential structure of a finite-dimensional vector space, we also have isomorphic Lie groups because we assumed the differential structure on $\mathcal{G}$ to be identical with the one induced from $\mathcal{A}$.

Let $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{A}$ be arbitrary vectors, $\boldsymbol{1} \in \mathcal{A}$ denote the multiplicative identity element, and Ψ the coordinate map $\mathcal{A} \to \mathbb{R}^{\dim\mathcal{A}}$ with respect to the default basis for $\mathcal{A}$, as before. We will now prove the injectivity and homomorphy of $\cdot^L$, which together implies it to be an monomorphism.

1) The injectivity basically follows from $\mathcal{A}$ having a one and Ψ being injective:

$$\boldsymbol{a}^L = \boldsymbol{b}^L \Rightarrow \boldsymbol{a}^L\Psi(\boldsymbol{1}) = \boldsymbol{b}^L\Psi(\boldsymbol{1}) \underset{\text{def}}{\Rightarrow} \Psi(\boldsymbol{a}\boldsymbol{1}) = \Psi(\boldsymbol{b}\boldsymbol{1}) \Rightarrow \boldsymbol{a}\boldsymbol{1} = \boldsymbol{b}\boldsymbol{1} \Rightarrow \boldsymbol{a} = \boldsymbol{b}$$

2) Algebra-homomorphy:
Let $\alpha \in \mathbb{R}$ be an arbitrary real number and $\boldsymbol{v} \in \mathcal{A}$ be a further arbitrary vector.

From the fact that $\mathcal{A}$'s vector multiplication, the matrix multiplication, and Ψ are (bi)linear follows then

$$(\boldsymbol{a} + \boldsymbol{b})^L\Psi(\boldsymbol{v}) \underset{\text{def}}{=} \Psi((\boldsymbol{a} + \boldsymbol{b})\boldsymbol{v}) = \Psi(\boldsymbol{a}\boldsymbol{v} + \boldsymbol{b}\boldsymbol{v}) \underset{\text{def}}{=} \boldsymbol{a}^L\Psi(\boldsymbol{v}) + \boldsymbol{b}^L\Psi(\boldsymbol{v}) = (\boldsymbol{a}^L + \boldsymbol{b}^L)\Psi(\boldsymbol{v}) \tag{D1}$$

and

$$(\alpha \boldsymbol{a})^L \Psi(\boldsymbol{v}) \underset{\text{def}}{=} \Psi((\alpha \boldsymbol{a})\boldsymbol{v}) = \Psi(\boldsymbol{a}(\alpha \boldsymbol{v})) \underset{\text{def}}{=} \boldsymbol{a}^L \Psi(\alpha \boldsymbol{v}) = \boldsymbol{a}^L(\alpha \Psi(\boldsymbol{v})) = (\alpha \boldsymbol{a}^L)\Psi(\boldsymbol{v}) \tag{D2}$$

From the associativity of both the $\mathcal{A}$-vector multiplication and the matrix product, it follows that

$$(\boldsymbol{ab})^L \Psi(\boldsymbol{v}) \underset{\text{def}}{=} \Psi((\boldsymbol{ab})\boldsymbol{v}) = \Psi(\boldsymbol{a}(\boldsymbol{bv})) \underset{\text{def}}{=} \boldsymbol{a}^L(\boldsymbol{b}^L \Psi(\boldsymbol{v})) = (\boldsymbol{a}^L \boldsymbol{b}^L)\Psi(\boldsymbol{v}) \tag{D3}$$

Because $\boldsymbol{v}$, and with it $\Psi(\boldsymbol{v})$, was arbitrary, it follows from the last three equations by the the fact that two square matrices are identical if they always yield the same when multiplied with arbitrary vectors that

$$(\boldsymbol{a} + \boldsymbol{b})^L = \boldsymbol{a}^L + \boldsymbol{b}^L, \qquad (\alpha \boldsymbol{a})^L = \alpha \boldsymbol{a}^L \quad \text{and} \quad (\boldsymbol{ab})^L = \boldsymbol{a}^L \boldsymbol{b}^L$$

Because $\boldsymbol{a}$ and $\boldsymbol{b}$ were arbitrary, it follows the homomorphy for the three algebra operations, the addition, and the scalar and vectorial multiplications. □

Appendix E: Proof of Theorem 2

Theorem 2: In our setting, $\exp_{\mathcal{G}}$ is a restriction of $\exp_{\mathcal{A}}$ to $\mathfrak{g} \subset \mathcal{A}$, employing the natural identification of $\mathcal{G}$'s tangent spaces with subvector spaces of $\mathcal{A}$.

Proof: It is a well known fact for matrix Lie groups that Theorem 2 holds for the pair $(\mathcal{G}^L, \mathbb{R}^{\dim \mathcal{A} \times \dim \mathcal{A}})$. The exponential map on $\mathcal{A}^L$ is obviously a restriction of the matrix exponential on $\mathbb{R}^{\dim \mathcal{A} \times \dim \mathcal{A}}$ to $\mathcal{A}^L$. Restricting $\exp_{\mathbb{R}^{\dim \mathcal{A} \times \dim \mathcal{A}}}$ first to $\mathcal{A}^L$ and then to $\mathcal{G}^L$ yields the same as restricting directly to $\mathcal{G}^L$, and therefore the statement follows for the pair $(\mathcal{G}^L, \mathcal{A}^L)$. Finally, because of Theorem 1, the statement also follows for the isomorphic pair $(\mathcal{G}, \mathcal{A})$ because both notions of exponential maps are intrinsically defined. □

Acknowledgment

This project has received funding from the European Union's Seventh Framework Programme for research, technological development, and demonstration under grant agreement 610603, EUROPA2.

References

[1] Strasdat, H., Montiel, J., and Davison, A. J., "Visual SLAM: Why Filter?" *Image and Vision Computing*, Vol. 30, No. 2, 2012, pp. 65–77. doi:10.1016/j.imavis.2012.02.009

[2] Leutenegger, S., Furgale, P., Rabaud, V., Chli, M., Konolige, K., and Siegwart, R., "Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization," *Robotics: Science and Systems IX*, edited by Newman, P., Fox, D., and Hsu, D., Paper 37, 2013, http://www.roboticsproceedings.org/rss09/.

[3] Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J., and Dellaert, F., "iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree," *International Journal of Robotics Research*, Vol. 31, No. 2, 2012, pp. 217–236. doi:10.1177/0278364911430419

[4] Vandersteen, J., Diehl, M., Aerts, C., and Swevers, J., "Spacecraft Attitude Estimation and Sensor Calibration Using Moving Horizon Estimation," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 3, 2013, pp. 734–742. doi:10.2514/1.58805

[5] Furgale, P. T., Barfoot, T. D., and Sibley, G., "Continuous-Time Batch Estimation Using Temporal Basis Functions," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, IEEE Publ., Piscataway, NJ, May 2012, pp. 2088–2095. doi:10.1109/ICRA.2012.6225005

[6] Wertz, J. R., *Spacecraft Attitude Determination and Control*, Kluwer Academic, Dordrecht, The Netherlands, 1978, pp. 426–428.

[7] Shuster, M. D., and Oh, S. D., "Three-Axis Attitude Determination from Vector Observations," *Journal of Guidance, Control, and Dynamics*, Vol. 4, No. 1, 1981, pp. 70–77. doi:10.2514/3.19717.

[8] Mortari, D., "ESOQ: A Closed-Form Solution to the Wahba Problem," *Journal of the Astronautical Sciences*, Vol. 45, No. 2, 1997, pp. 195–204.

[9] Mortari, D., "Second Estimator of the Optimal Quaternion," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 5, 2000, pp. 885–888. doi:10.2514/2.4618

[10] Farrell, J. L., "Attitude Determination by Kalman Filtering," *Automatica*, Vol. 6, No. 3, 1970, pp. 419–430. doi:10.1016/0005-1098(70)90057-9

[11] Lefferts, E. J., Markley, F. L., and Shuster, M. D., "Kalman Filtering for Spacecraft Attitude Estimation," *Journal of Guidance, Control, and Dynamics*, Vol. 5, No. 5, 1982, pp. 417–429. doi:10.2514/3.56190

[12] Bar-Itzhack, I. Y., and Oshman, Y., "Attitude Determination from Vector Observations: Quaternion Estimation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 21, No. 1, 1985, pp. 128–136. doi:10.1109/TAES.1985.310546

[13] Shuster, M. D., "A Simple Kalman Filter and Smoother for Spacecraft Attitude," *Journal of the Astronautical Sciences*, Vol. 37, No. 1, 1989, pp. 89–106.

[14] Crassidis, J. L., and Markley, F. L., "Unscented Filtering for Spacecraft Attitude Estimation," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 4, 2003, pp. 536–542. doi:10.2514/2.5102

[15] Bauchau, O., and Trainelli, L., "The Vectorial Parameterization of Rotation," *Nonlinear Dynamics*, Vol. 32, No. 1, 2003, pp. 71–92. doi:10.1023/A:1024265401576

[16] Hughes, P. C., *Spacecraft Attitude Dynamics*, Wiley, New York, 1986, pp. 30–31.

[17] Kim, M.-J., Kim, M.-S., and Shin, S. Y., "A General Construction Scheme for Unit Quaternion Curves with Simple High Order Derivatives," *SIGGRAPH '95 Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, Association for Computing Machinery, New York, 1995, pp. 369–376. doi:10.1145/218380.218486

[18] Mourikis, A. I., Trawny, N., Roumeliotis, S. I., Johnson, A. E., Ansar, A., and Matthies, L., "Vision-Aided Inertial Navigation for Spacecraft Entry, Descent, and Landing," *IEEE Transactions on Robotics*, Vol. 25, No. 2, 2009, pp. 264–280. doi:10.1109/TRO.2009.2012342

[19] Li, M., and Mourikis, A. I., "High-Precision, Consistent EKF-Based Visual-Inertial Odometry," *International Journal of Robotics Research*, Vol. 32, No. 6, 2013, pp. 690–711. doi:10.1177/0278364913481251

[20] Moore, J. B., and Tam, P. K., "Fixed-Lag Smoothing for Nonlinear Systems with Discrete Measurements," *Information Sciences*, Vol. 6, Jan. 1973, pp. 151–160. doi:10.1016/0020-0255(73)90032-7

[21] Poli, D., and Toutin, T., "Review of Developments in for High Resolution Satellite Pushbroom Sensors," *Photogrammetric Record*, Vol. 27, No. 137, 2012, pp. 58–73. doi:10.1111/j.1477-9730.2011.00665.x

[22] Ringaby, E., and Forssén, P.-E., "Efficient Video Rectification and Stabilisation for Cell-Phones," *International Journal of Computer Vision*, Vol. 96, No. 3, 2011, pp. 335–352. doi:10.1007/s11263-011-0465-8

[23] Hedborg, J., Forssen, P.-E., Felsberg, M., and Ringaby, E., "Rolling Shutter Bundle Adjustment," *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Publ., Piscataway, NJ, pp. 1434–1441. doi:10.1109/CVPR.2012.6247831

[24] Oth, L., Furgale, P., Kneip, L., and Siegwart, R., "Rolling Shutter Camera Calibration," *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Publ., Piscataway, NJ, pp. 1360–1367. doi:10.1109/CVPR.2013.179

[25] Bosse, M., Zlot, R., and Flick, P., "Zebedee: Design of a Spring-Mounted 3-D Range Sensor with Application to Mobile Mapping," *IEEE Transactions on Robotics*, Vol. 28, No. 5, 2012, pp. 1104–1119. doi:10.1109/TRO.2012.2200990

[26] Dong, H. J., and Barfoot, T. D., "Lighting-Invariant Visual Odometry Using Lidar Intensity Imagery and Pose Interpolation" *Proceedings of the International Conference on Field and Service Robotics (FSR)*, Springer, Berlin, 2014, pp. 327–342. doi:10.1007/978-3-642-40686-7_22

[27] Anderson, S., and Barfoot, T. D., "Towards Relative Continuous-Time SLAM," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, IEEE Publ., Piscataway, NJ, pp. 1033–1040. doi:10.1109/ICRA.2013.6630700

[28] Tong, C. H., and Barfoot, T. D., "Gaussian Process Gauss–Newton for 3D Laser-Based Visual Odometry," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, IEEE Publ., Piscataway, NJ, pp. 5204–5211. doi:10.1109/ICRA.2013.6631321

[29] Tong, C. H., Furgale, P. T., and Barfoot, T. D., "Gaussian Process Gauss–Newton for Non-Parametric Simultaneous Localization and Mapping," *International Journal of Robotics Research*, Vol. 32, No. 5, 2013, pp. 507–525. doi:10.1177/0278364913478672

[30] Jazwinski, A. H., *Stochastic Processes and Filtering Theory*, Academic Press, New York, 1970, Chap. 5.

[31] Furgale, P., Rehder, J., and Siegwart, R., "Unified Temporal and Spatial Calibration for Multi-Sensor Systems," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE Publ., Piscataway, NJ, 2013, pp. 1280–1286. doi:10.1109/IROS.2013.6696514

[32] Lovegrove, S., Patron-Perez, A., and Sibley, G., "Spline Fusion: A Continuous-Time Representation for Visual-Inertial Fusion with Application to Rolling Shutter Cameras," *Proceedings of the British Machine Vision Conference*, British Machine Vision Assoc. Press, Durham, 2013, pp. 93.1–93.12. doi:10.5244/C.27.93

[33] Howard, T., and Kelly, A., "Optimal Rough Terrain Trajectory Generation for Wheeled Mobile Robots," *International Journal of Robotics Research*, Vol. 26, No. 2, 2007, pp. 141–166. doi:10.1177/0278364906075328

[34] Wahba, G., "Problem 65-1: A Least-Squares Estimate of Satellite Attitude," *SIAM Review*, Vol. 7, No. 3, 1965, p. 409. doi:10.1137/1007077

[35] Farrel, J. L., Stuelpnagel, J. C., Wessner, R. H., Velman, J. R., and Brook, J. E., "A Least-Squares Estimate of Satellite Attitude," *SIAM Review*, Vol. 8, No. 3, 1966, pp. 384–386. doi:10.1137/1008080

[36] Markley, F. L., "Attitude Determination Using Vector Observations and the Singular Value Decomposition," *Journal of the Astronautical Sciences*, Vol. 36, No. 3, 1988, pp. 245–258.

[37] Forbes, J. R., and de Ruiter, A. H. J., "LMI-Based Solution to Wahba's Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 38, No. 1, 2014, pp. 147–151. doi:10.2514/1.G000132

[38] de Ruiter, A. H. J., and Forbes, J. R., "On the Solution of Wahba's Problem on SO(n)" *Journal of the Astronautical Sciences*, Vol. 60, No. 1, 2013, pp. 1–31. doi:10.1007/s40295-014-0019-8

[39] Crassidis, J. L., Markley, F. L., and Cheng, Y., "A Survey of Nonlinear Attitude Estimation Methods," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 1, 2007, pp. 12–28. doi:10.2514/1.22452

[40] Bar-Itzhack, I. Y., and Reiner, J., "Recursive Attitude Determination from Vector Observations: Direction Cosine Matrix Identification," *Journal of Guidance, Control, and Dynamics*, Vol. 7, No. 1, 1984, pp. 51–56. doi:10.2514/3.56362

[41] Choukroun, D., Bar-Itzhack, I. Y., and Oshman, Y., "A Novel Quaternion Kalman Filter," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 42, No. 1, 2006, pp. 174–190. doi:10.1109/TAES.2006.1603413

[42] Markley, F. L., "Attitude Error Representations for Kalman Filtering," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 2, 2003, pp. 311–317. doi:10.2514/2.5048

[43] Markley, F. L., "Multiplicative vs. Additive Filtering for Spacecraft Attitude Determination," *Proceedings of the 6th Conference on Dynamics and Control of Systems and Structures in Space (DCSSS)*, Paper D22, July 2014, http://naca.central.cranfield.ac.uk/dcsss/2004/index.html.

[44] Zanetti, R., Majji, M., Bishop, R., and Mortari, D., "Norm-Constrained Kalman Filtering," *Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 5, 2009, pp. 1458–1465. doi:10.2514/1.G000344

[45] Chee, S. A., and Forbes, J. R., "Norm-Constrained Consider Kalman Filtering," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 6, 2014, pp. 2048–2053. doi:10.2514/1.G000344

[46] Forbes, J. R., de Ruiter, A. H. J., and Zlotnik, D. E., "Continuous-Time Norm-Constrained Kalman Filtering," *Automatica*, Vol. 50, No. 10, 2014, pp. 2546–2554. doi:10.1016/j.automatica.2014.08.007

[47] Choukroun, D., Weiss, H., Bar-Itzhack, I. Y., and Oshman, Y., "Kalman Filtering for Matrix Estimation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 42, No. 1, 2006, pp. 147–159. doi:10.1109/TAES.2006.1603411

[48] Choukroun, D., Weiss, H., Bar-Itzhack, I. Y., and Oshman, Y., "Direction Cosine Matrix Estimation from Vector Observations Using a Matrix Kalman Filter," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 46, No. 1, 2010, pp. 61–79. doi:10.1109/TAES.2010.5417148

[49] Khosravian, A., and Namvar, M., "Rigid Body Attitude Control Using a Single Vector Measurement and Gyro," *IEEE Transactions on Automatic Control*, Vol. 57, No. 5, 2012, pp. 1273–1279. doi:10.1109/TAC.2011.2174663

[50] Firoozi, D., and Namvar, M., "Analysis of Gyro Noise in Non-Linear Attitude Estimation Using a Single Vector Measurement," *IET Control Theory and Applications*, Vol. 6, No. 14, 2012, pp. 2226–2234. doi:10.1049/iet-cta.2011.0347

[51] Mahony, R., Hamel, T., and Pflimlin, J.-M., "Complementary Filter Design on the Special Orthogonal Group SO(3)," *Proceedings of the 44th IEEE Conference on Decision and Control and the European Control Conference*, IEEE Publ., Piscataway, NJ, Dec. 2005, pp. 1477–1484. doi:10.1109/CDC.2005.1582367

[52] Hamel, T., and Mahony, R., "Attitude Estimation on SO(3) Based on Direct Inertial Measurements," *Proceedings of the IEEE Conference on Robotics and Automation*, IEEE Publ., Piscataway, NJ, May 2006, pp. 2170–2175. doi:10.1109/ROBOT.2006.1642025

[53] Mahony, R., Hamel, T., and Pflimlin, J.-M., "Nonlinear Complementary Filters on the Special Orthogonal Group," *IEEE Transactions on Automatic Control*, Vol. 53, No. 5, 2008, pp. 1203–1218, doi:10.1109/TAC.2008.923738

[54] Jensen, K. J., "Generalized Nonlinear Complementary Attitude Filter," *Journal of Guidance, Control, and Dynamics*, Vol. 34, No. 5, 2011, pp. 1588–1592. doi:10.2514/1.53467

[55] Kinsey, J. C., and Whitcomb, L. L., "Adaptive Identification on the Group of Rigid-Body Rotations and Its Application to Underwater Vehicle Navigation," *IEEE Transactions on Robotics*, Vol. 23, No. 1, 2007, pp. 124–136. doi:10.1109/TRO.2006.886829

[56] Grip, H. F., Fossen, T., Johansen, T. A., and Saberi, A., "Attitude Estimation Using Biased Gyro and Vector Measurements with Time-Varying Reference Vectors," *IEEE Transactions on Automatic Control*, Vol. 57, No. 5, 2012, pp. 1332–1338. doi:10.1109/TAC.2011.2173415

[57] Bartels, R. H., Beatty, J. C., and Barsky, B. A., *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann, Los Altos, CA, 1987, Chap. 4.

[58] Kirillov, A. A., Kirillov, A. A., and Kirillov, A. A., *An Introduction to Lie Groups and Lie Algebras*, Vol. 113, Cambridge Studies in Advanced Mathematics, Cambridge Univ. Press, Cambridge, England, U.K., 2008, Chaps. 2–3. doi:10.1017/CBO9780511755156

[59] Hall, B., *Lie Groups, Lie Algebras, and Representations: An Elementary Introduction*, Vol. 222, Springer, Berlin, 2003, Chaps. 1–3. doi:10.1007/978-0-387-21554-9

[60] Rasmussen, C. E., and Williams, C., *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, MA, 2006, Chap. 2.

[61] de Boor, C., *A Practical Guide to Splines*, Springer–Verlag, New York, 2001, Chaps. 9–10, 14. doi:10.1137/1022106

[62] Park, F. C., and Ravani, B., "Smooth Invariant Interpolation of Rotations," *ACM Transactions on Graphics*, Vol. 16, No. 3, 1997, pp. 277–295. doi:10.1145/256157.256160

[63] Barfoot, T. D., Forbes, J. R., and Furgale, P. T., "Pose Estimation Using Linearized Rotations and Quaternion Algebra," *Acta Astronautica*, Vol. 68, Nos. 1–2, 2011, pp. 101–112. doi:10.1016/j.actaastro.2010.06.049

[64] Kim, M.-S., and Nam, K.-W., "Interpolating Solid Orientations with Circular Blending Quaternion Curves," *Computer Aided Design*, Vol. 27, No. 5, 1995, pp. 385–398. doi:10.1016/0010-4485(95)96802-S

[65] Mirzaei, F., and Roumeliotis, S., "A Kalman Filter-Based Algorithm for IMU-Camera Calibration: Observability Analysis and Performance Evaluation," *IEEE Transactions on Robotics*, Vol. 24, No. 5, 2008, pp. 1143–1156. doi:10.1109/TRO.2008.2004486

[66] Kelly, J., and Sukhatme, G. S., "Visual-Inertial Sensor Fusion: Localization, Mapping and Sensor-to-Sensor Self-Calibration," *International Journal of Robotics Research*, Vol. 30, No. 1, 2011, pp. 56–79. doi:10.1177/0278364910382802