Acta Astronautica 68 (2011) 101–112

Contents lists available at ScienceDirect

Acta Astronautica

journal homepage: www.elsevier.com/locate/actaastro

Pose estimation using linearized rotations and quaternion algebra

Timothy Barfoot *, James R. Forbes, Paul T. Furgale

University of Toronto, Institute for Aerospace Studies, 4925 Dufferin Street, Toronto, Ontario, Canada M3H 5T6

ARTICLE INFO

Article history:
Received 27 April 2010
Accepted 30 June 2010
Available online 4 August 2010

Keywords:
Pose estimation
Linearized rotations
Quaternion algebra

ABSTRACT

In this paper we revisit the topic of how to formulate error terms for estimation problems that involve rotational state variables. We present a first-principles linearization approach that yields multiplicative error terms for unit-length quaternion representations of rotations, as well as for canonical rotation matrices. Quaternion algebra is employed throughout our derivations. We show the utility of our approach through two examples: (i) linearizing a sun sensor measurement error term, and (ii) weighted-least-squares point-cloud alignment.

1. Introduction

The classic results of multivariate state estimation are rooted in both probability theory and vector calculus. This presents challenges for many practical estimation problems involving rotational state variables, which are not members of a vector space. Rather, the set of rotations constitutes a *non-commutative group*, called $SO(3)$. Regardless of the choice of representation (e.g., rotation matrix, unit-length quaternion, Euler angles), a rotation has exactly three degrees of freedom. All rotational representations involving exactly three parameters have singularities [27] and all representations having more than three parameters have constraints. The question of how best to parametrize and handle rotations in state estimation is by no means new. There are many rotational parameterizations available, each with its unique advantages and disadvantages [22]. In spacecraft attitude and robotics estimation, the 4×1 unit-length quaternion (a.k.a., Euler–Rodrigues symmetric parameters), the standard 3×3 rotation matrix, and Euler angles are all common [4].

Unit-length quaternions are appealing in that they are free of singularities and compact in their representation; however, the unit-length constraint must be considered carefully, particularly for estimation problems. The question of whether to make updates to unit-length quaternions 'additively' or 'multiplicatively' has been debated often [17,16]. For online attitude estimation, two variants of the Kalman filter [11] have been introduced: The Additive Extended Kalman Filter (AEKF) [1] and the Multiplicative Extended Kalman Filter (MEKF) [12,15]. Both of these require the estimated quaternion to be re-normalized after the filter update step to restore the unit-length constraint. However, it was only recently shown that re-normalization within the Kalman filter framework is in fact a direct consequence of performing optimal estimation under the unit-length constraint [32]. More generally, Shuster [23,24] provides a detailed examination of performing both 'constrained' (e.g., sensitive to but not enforcing the unit-length quaternion constraint during the update step) and 'unconstrained' (e.g., not sensitive to the unit-length quaternion constraint), demonstrating that constraint-sensitive estimation is preferable.

This paper builds on the work of Shuster [23,24], by deriving constraint-sensitive linearizations of both unit-length quaternions and rotation matrices from a simple first-principles linearization perspective. Under this approach, the multiplicative perturbations arise quite naturally. We show how these perturbations may be used in the calculation of an arbitrary 'measurement sensitivity matrix' (i.e., partial derivative of the measurement function with respect to the rotation) [23] and more generally in the linearization of any expression involving

* Corresponding author. Tel.: +1 416 667 7719; fax: +1 416 667 7799.
E-mail address: tim.barfoot@utoronto.ca (T. Barfoot).

0094-5765/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.actaastro.2010.06.049

a unit-length quaternion or rotation matrix. In our handling of unit-length quaternions, we exploit 'quaternion algebra' [10] quite heavily, which permits parallels to be drawn between the unit-length quaternion and rotation matrix results. To demonstrate the utility of our approach, we provide an example of linearizing a sun sensor measurement error term as well as an extended example of a batch weighted-least-squares pose (i.e., position and orientation) estimation problem. Pose estimation is an important problem in such aerospace applications as spacecraft rendezvous and docking, as well as motion estimation of planetary rovers.

The paper is organized as follows. We first introduce the notation and operations we will use for handling quaternions. We then derive our general expressions for linearizing expressions involving both unit-length quaternions and rotation matrices. Finally, we demonstrate these through the sun sensor and weighted-least-squares examples.

2. Quaternion algebra

In this section we introduce the notation that we will employ in our treatment of quaternions [6,28,7], which we refer to as *quaternion algebra* [10]. We use quaternions to represent both translations and rotations. Our notation differs slightly from others [3,22], but the concepts remain the same. We begin with notation followed by identities and finally the relationship to rotations (i.e., unit-length quaternions).

2.1. Notation

In what is to follow, a *quaternion* will be a 4×1 column that may be written as

$$\mathbf{q} := \begin{bmatrix} \boldsymbol{\varepsilon} \\ \eta \end{bmatrix}, \tag{1}$$

where $\boldsymbol{\varepsilon}$ is a 3×1 and η is a scalar. The quaternion *left-hand compound* operator, $+$, and the *right-hand compound* operator, $\oplus$, will be defined as

$$\mathbf{q}^{+} := \begin{bmatrix} \eta\mathbf{1}-\boldsymbol{\varepsilon}^{\times} & \boldsymbol{\varepsilon} \\ -\boldsymbol{\varepsilon}^T & \eta \end{bmatrix} \quad \text{and} \quad \mathbf{q}^{\oplus} := \begin{bmatrix} \eta\mathbf{1}+\boldsymbol{\varepsilon}^{\times} & \boldsymbol{\varepsilon} \\ -\boldsymbol{\varepsilon}^T & \eta \end{bmatrix}, \tag{2}$$

where

$$\boldsymbol{\varepsilon}^{\times} := \begin{bmatrix} 0 & -\varepsilon_3 & \varepsilon_2 \\ \varepsilon_3 & 0 & -\varepsilon_1 \\ -\varepsilon_2 & \varepsilon_1 & 0 \end{bmatrix} \tag{3}$$

defines the usual 3×3 skew-symmetric matrix, which may be used to implement the cross product for 3×1 columns [9]. Under these definitions, the *multiplication* of quaternions, $\mathbf{u}$ and $\mathbf{v}$, which is typically written as $\mathbf{u} \otimes \mathbf{v}$ [22], may be written equivalently as either

$$\mathbf{u}^{+}\mathbf{v} \quad \text{or} \quad \mathbf{v}^{\oplus}\mathbf{u}, \tag{4}$$

which are both products of a 4×4 matrix with a 4×1 column. The *conjugate* operator for quaternions, -1, will be defined by

$$\mathbf{q}^{-1} := \begin{bmatrix} -\boldsymbol{\varepsilon} \\ \eta \end{bmatrix}. \tag{5}$$

According to Whittaker [31], the scalar formulae for the multiplication of quaternions were independently discovered by Gauss, Rodrigues, Hamilton, and Cayley at different times. They were first written as a matrix multiplication by Peirce [21] and later in the spacecraft attitude dynamics field by Ickes [10], according to Ell [5]. Note that we have not yet restricted ourselves to unit-length quaternions, which may be used to represent rotations.

2.2. Identities

Let $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{w}$ be quaternions. Then some useful identities are

$$\mathbf{u}^{+}\mathbf{v}^{\oplus} \equiv \mathbf{v}^{\oplus}\mathbf{u}^{+}, \tag{6}$$

and

$$(\mathbf{u}^{+})^T \equiv (\mathbf{u}^{+})^{-1} \equiv (\mathbf{u}^{-1})^{+}, \quad (\mathbf{u}^{\oplus})^T \equiv (\mathbf{u}^{\oplus})^{-1} \equiv (\mathbf{u}^{-1})^{\oplus},$$

$$(\mathbf{u}^{+}\mathbf{v})^{-1} \equiv \mathbf{v}^{-1^{+}}\mathbf{u}^{-1}, \quad (\mathbf{u}^{\oplus}\mathbf{v})^{-1} \equiv \mathbf{v}^{-1^{\oplus}}\mathbf{u}^{-1},$$

$$(\mathbf{u}^{+}\mathbf{v})^{+}\mathbf{w} \equiv \mathbf{u}^{+}(\mathbf{v}^{+}\mathbf{w}) \equiv \mathbf{u}^{+}\mathbf{v}^{+}\mathbf{w},$$

$$(\mathbf{u}^{\oplus}\mathbf{v})^{\oplus}\mathbf{w} \equiv \mathbf{u}^{\oplus}(\mathbf{v}^{\oplus}\mathbf{w}) \equiv \mathbf{u}^{\oplus}\mathbf{v}^{\oplus}\mathbf{w},$$

$$\alpha\mathbf{u}^{+} + \beta\mathbf{v}^{+} \equiv (\alpha\mathbf{u}+\beta\mathbf{v})^{+}, \quad \alpha\mathbf{u}^{\oplus} + \beta\mathbf{v}^{\oplus} \equiv (\alpha\mathbf{u}+\beta\mathbf{v})^{\oplus}, \tag{7}$$

where α and β are scalars. Most of these can be found in [3,22]. We omit the proofs in the interest of space.

The set of quaternions forms a *non-commutative group* under both the $+$ and $\oplus$ operations [22]. Many of the identities above are prerequisites to showing this fact. The *identity element* of this group, $\boldsymbol{\iota} := [0\ 0\ 0\ 1]^T$, is such that

$$\boldsymbol{\iota}^{+} = \boldsymbol{\iota}^{\oplus} = \mathbf{1}, \tag{8}$$

where $\mathbf{1}$ is the 4×4 identity matrix.

2.3. Relationship to rotations and translations

Rotations may be represented in this notation by using a unit-length quaternion, $\mathbf{q}$. The unit-length constraint can be written as

$$\mathbf{q}^T\mathbf{q} = 1. \tag{9}$$

The set of unit-length quaternions forms a *non-commutative sub-group* of the quaternions known as *SO*(3). A point, $\mathbf{v}$ (or a translation), may be represented by a quaternion of the form

$$\mathbf{v} = \begin{bmatrix} \boldsymbol{\rho} \\ 0 \end{bmatrix}, \tag{10}$$

where $\boldsymbol{\rho} = [x\ y\ z]^T$ is the usual 3×1 expression. To rotate a point, $\mathbf{v}$, to a new frame using the rotation, $\mathbf{q}$, we compute

$$\mathbf{u} = \mathbf{q}^{+}\mathbf{v}^{+}\mathbf{q}^{-1} = \mathbf{q}^{+}\mathbf{q}^{-1^{\oplus}}\mathbf{v} = \mathbf{R}\mathbf{v}, \tag{11}$$

where

$$\mathbf{R} := \mathbf{q}^{+}\mathbf{q}^{-1^{\oplus}} = \mathbf{q}^{-1^{\oplus}}\mathbf{q}^{+} = \mathbf{q}^{\oplus^T}\mathbf{q}^{+} = \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{12}$$

and $\mathbf{C}$ is the canonical 3×3 rotation matrix [22] representing the same rotation as $\mathbf{q}$. It follows that:

$$\mathbf{u} = \begin{bmatrix} \mathbf{C}\boldsymbol{\rho} \\ 0 \end{bmatrix}, \tag{13}$$

as expected. It is worth noting that for unit-length quaternions, we will refer to the conjugate as the *inverse*.

3. Linearizing expressions involving rotations

In this section we present our main result, which is a re-examination of how to linearize expressions involving rotations. We treat both canonical rotation matrices as well as unit-length quaternions. Our approach is a simple first-principles Taylor approximation. We will see that the so-called multiplicative perturbations arise quite naturally. In the interest of accessibility, we begin with the rotation matrix case, followed by unit-length quaternions, and then show how to linearize a typical sun sensor measurement error term using our derived expressions.

3.1. Rotation matrix approach

To begin, we require the establishment of two identities. Euler's theorem allows us to write a rotation matrix, $\mathbf{C}$, in terms of a rotation about a unit-length axis, $\mathbf{a}$, through an angle, φ [9]:

$$\mathbf{C}(\mathbf{a},\varphi) = \cos\varphi\mathbf{1} + (1-\cos\varphi)\mathbf{a}\mathbf{a}^T - \sin\varphi\mathbf{a}^{\times}. \tag{14}$$

We may now take the partial derivative of $\mathbf{C}(\mathbf{a},\varphi)$ with respect to the angle, φ:

$$\frac{\partial\mathbf{C}(\mathbf{a},\varphi)}{\partial\varphi} = -\sin\varphi\mathbf{1} + \sin\varphi\mathbf{a}\mathbf{a}^T - \cos\varphi\mathbf{a}^{\times} \tag{15a}$$

$$= \sin\varphi\underbrace{(-\mathbf{1}+\mathbf{a}\mathbf{a}^T)}_{\mathbf{a}^{\times}\mathbf{a}^{\times}} - \cos\varphi\mathbf{a}^{\times} \tag{15b}$$

$$= -\cos\varphi\mathbf{a}^{\times} - (1-\cos\varphi)\underbrace{\mathbf{a}^{\times}\mathbf{a}}_{\mathbf{0}}\mathbf{a}^T + \sin\varphi\mathbf{a}^{\times}\mathbf{a}^{\times} \tag{15c}$$

$$= -\mathbf{a}^{\times}\underbrace{(\cos\varphi\mathbf{1} + (1-\cos\varphi)\mathbf{a}\mathbf{a}^T - \sin\varphi\mathbf{a}^{\times})}_{\mathbf{C}(\mathbf{a},\varphi)}. \tag{15d}$$

Thus, our first key identity is

$$\frac{\partial\mathbf{C}(\mathbf{a},\varphi)}{\partial\varphi} \equiv -\mathbf{a}^{\times}\mathbf{C}(\mathbf{a},\varphi). \tag{16}$$

An immediate application of this is that for any principal-axis rotation, $\mathbf{C}_{\psi}(\theta)$, about principal axis ψ and through angle θ, we have

$$\frac{\partial\mathbf{C}_{\psi}(\theta)}{\partial\theta} = -\mathbf{1}_{\psi}^{\times}\mathbf{C}_{\psi}(\theta), \tag{17}$$

where $\mathbf{1}_{\psi}$ is column ψ of the 3×3 identity matrix. Let us now consider an α–β–γ Euler sequence (with $\alpha\neq\beta$ and $\beta\neq\gamma$),

$$\mathbf{C}(\boldsymbol{\theta}) := \mathbf{C}_{\gamma}(\theta_3)\mathbf{C}_{\beta}(\theta_2)\mathbf{C}_{\alpha}(\theta_1), \tag{18}$$

where $\boldsymbol{\theta} := [\theta_1\ \theta_2\ \theta_3]^T$. Furthermore, select an arbitrary constant 3×1 column, $\mathbf{v}$. Applying (17), we have

$$\frac{\partial(\mathbf{C}(\boldsymbol{\theta})\mathbf{v})}{\partial\theta_3} = -\mathbf{1}_{\gamma}^{\times}\mathbf{C}_{\gamma}(\theta_3)\mathbf{C}_{\beta}(\theta_2)\mathbf{C}_{\alpha}(\theta_1)\mathbf{v} = (\mathbf{C}(\boldsymbol{\theta})\mathbf{v})^{\times}\mathbf{1}_{\gamma}, \tag{19a}$$

$$\frac{\partial(\mathbf{C}(\boldsymbol{\theta})\mathbf{v})}{\partial\theta_2} = -\mathbf{C}_{\gamma}(\theta_3)\mathbf{1}_{\beta}^{\times}\mathbf{C}_{\beta}(\theta_2)\mathbf{C}_{\alpha}(\theta_1)\mathbf{v} = (\mathbf{C}(\boldsymbol{\theta})\mathbf{v})^{\times}\mathbf{C}_{\gamma}(\theta_3)\mathbf{1}_{\beta}, \tag{19b}$$

$$\frac{\partial(\mathbf{C}(\boldsymbol{\theta})\mathbf{v})}{\partial\theta_1} = -\mathbf{C}_{\gamma}(\theta_3)\mathbf{C}_{\beta}(\theta_2)\mathbf{1}_{\alpha}^{\times}\mathbf{C}_{\alpha}(\theta_1)\mathbf{v} = (\mathbf{C}(\boldsymbol{\theta})\mathbf{v})^{\times}\mathbf{C}_{\gamma}(\theta_3)\mathbf{C}_{\beta}(\theta_2)\mathbf{1}_{\alpha}, \tag{19c}$$

where we have made use of the two general identities,

$$\mathbf{r}^{\times}\mathbf{s} \equiv -\mathbf{s}^{\times}\mathbf{r}, \tag{20a}$$

$$(\mathbf{C}\mathbf{s})^{\times} \equiv \mathbf{C}\mathbf{s}^{\times}\mathbf{C}^T \tag{20b}$$

for any 3×1 columns $\mathbf{r}$, $\mathbf{s}$ and any 3×3 rotation matrix, $\mathbf{C}$ [9]. Combining the results in (19) we have

$$\frac{\partial(\mathbf{C}(\boldsymbol{\theta})\mathbf{v})}{\partial\boldsymbol{\theta}} = \begin{bmatrix} \frac{\partial(\mathbf{C}(\boldsymbol{\theta})\mathbf{v})}{\partial\theta_1} & \frac{\partial(\mathbf{C}(\boldsymbol{\theta})\mathbf{v})}{\partial\theta_2} & \frac{\partial(\mathbf{C}(\boldsymbol{\theta})\mathbf{v})}{\partial\theta_3} \end{bmatrix}$$
$$= (\mathbf{C}(\boldsymbol{\theta})\mathbf{v})^{\times}\underbrace{[\mathbf{C}_{\gamma}(\theta_3)\mathbf{C}_{\beta}(\theta_2)\mathbf{1}_{\alpha}\ \ \mathbf{C}_{\gamma}(\theta_3)\mathbf{1}_{\beta}\ \ \mathbf{1}_{\gamma}]}_{\mathbf{S}(\boldsymbol{\theta})}, \tag{21}$$

and thus our second key identity is

$$\frac{\partial(\mathbf{C}(\boldsymbol{\theta})\mathbf{v})}{\partial\boldsymbol{\theta}} \equiv (\mathbf{C}(\boldsymbol{\theta})\mathbf{v})^{\times}\mathbf{S}(\boldsymbol{\theta}), \tag{22}$$

which we note is true regardless of the choice of Euler sequence. The matrix, $\mathbf{S}(\boldsymbol{\theta})$, is the usual matrix relating angular velocity to Euler-angle rates [9].

Having established identities (16) and (22), we now return to first principles and consider carefully how to linearize a rotation. If we have a function, $\mathbf{f}(\mathbf{x})$, of some variable, $\mathbf{x}$, then perturbing $\mathbf{x}$ slightly from its nominal value, $\bar{\mathbf{x}}$, by an amount $\delta\mathbf{x}$ will result in a change in the function. We can express this in terms of a Taylor-series expansion of $\mathbf{f}$ about $\bar{\mathbf{x}}$:

$$\mathbf{f}(\bar{\mathbf{x}}+\delta\mathbf{x}) = \mathbf{f}(\bar{\mathbf{x}}) + \left.\frac{\partial\mathbf{f}(\mathbf{x})}{\partial\mathbf{x}}\right|_{\bar{\mathbf{x}}}\delta\mathbf{x} + (\text{higher order terms}). \tag{23}$$

This presupposes that $\delta\mathbf{x}$ is not constrained in any way. The trouble with carrying out the same process with rotations is that most of the representations involve constraints and thus are not easily perturbed (without enforcing the constraint). The notable exceptions are the three-parameter representations, the most common of which are the Euler angle sequences. These contain exactly three parameters and thus each can be varied independently. For this reason, we choose to use Euler angles in our perturbation of functions involving rotations.

Consider perturbing $\mathbf{C}(\boldsymbol{\theta})\mathbf{v}$ with respect to Euler angles $\boldsymbol{\theta}$, where $\mathbf{v}$ is an arbitrary constant 3×1 column. Letting $\bar{\boldsymbol{\theta}} := [\bar{\theta}_1\ \bar{\theta}_2\ \bar{\theta}_3]^T$ and $\delta\boldsymbol{\theta} := [\delta\theta_1\ \delta\theta_2\ \delta\theta_3]^T$, then applying a first-order Taylor-series approximation we have

$$\mathbf{C}(\bar{\boldsymbol{\theta}}+\delta\boldsymbol{\theta})\mathbf{v} \approx \mathbf{C}(\bar{\boldsymbol{\theta}})\mathbf{v} + \left.\frac{\partial(\mathbf{C}(\boldsymbol{\theta})\mathbf{v})}{\partial\boldsymbol{\theta}}\right|_{\bar{\boldsymbol{\theta}}}\delta\boldsymbol{\theta} \tag{24a}$$

$$= \mathbf{C}(\overline{\boldsymbol\theta})\mathbf{v} + ((\mathbf{C}(\boldsymbol\theta)\mathbf{v})^{\times}\mathbf{S}(\boldsymbol\theta))|_{\overline{\boldsymbol\theta}}\delta\boldsymbol\theta \tag{24b}$$

$$= \mathbf{C}(\overline{\boldsymbol\theta})\mathbf{v} + (\mathbf{C}(\overline{\boldsymbol\theta})\mathbf{v})^{\times}\mathbf{S}(\overline{\boldsymbol\theta})\delta\boldsymbol\theta \tag{24c}$$

$$= \mathbf{C}(\overline{\boldsymbol\theta})\mathbf{v} - (\mathbf{S}(\overline{\boldsymbol\theta})\delta\boldsymbol\theta)^{\times}(\mathbf{C}(\overline{\boldsymbol\theta})\mathbf{v}) \tag{24d}$$

$$= \left(\mathbf{1} - \left(\mathbf{S}(\overline{\boldsymbol\theta})\delta\boldsymbol\theta\right)^{\times}\right)\mathbf{C}(\overline{\boldsymbol\theta})\mathbf{v}, \tag{24e}$$

where we have used (22) to get to the second line. Observing that $\mathbf{v}$ is arbitrary, we can drop it from both sides and write

$$\mathbf{C}(\overline{\boldsymbol\theta}+\delta\boldsymbol\theta) \approx \underbrace{\left(\mathbf{1} - \left(\mathbf{S}(\overline{\boldsymbol\theta})\delta\boldsymbol\theta\right)^{\times}\right)}_{\text{infinitesimal rot.}}\mathbf{C}(\overline{\boldsymbol\theta}), \tag{25}$$

which we see is the product of an infinitesimal rotation matrix [9] and the unperturbed rotation matrix, $\mathbf{C}(\overline{\boldsymbol\theta})$. It is worth noting that we did not assume the perturbation is of this multiplicative form, but rather showed that it is a consequence of the linearization procedure. Notationally, it is simpler to write

$$\mathbf{C}(\overline{\boldsymbol\theta}+\delta\boldsymbol\theta) \approx (\mathbf{1} - \delta\boldsymbol\phi^{\times})\mathbf{C}(\overline{\boldsymbol\theta}), \tag{26}$$

with $\delta\boldsymbol\phi := \mathbf{S}(\overline{\boldsymbol\theta})\delta\boldsymbol\theta$. Eq. (26) is revealing as it tells us how to perturb a rotation matrix when it appears inside any function. This may be done either in terms of perturbations to the Euler angles, $\delta\boldsymbol\theta$, or directly through the *rotation vector*, $\delta\boldsymbol\phi$.

Rotation matrices fundamentally have three degrees of freedom but are represented by nine parameters. There are therefore six constraints, which may be written as a single matrix orthogonality constraint: $\mathbf{C}\mathbf{C}^T = \mathbf{1}$. Suppose this constraint holds for $\mathbf{C}(\overline{\boldsymbol\theta})$. Then for the perturbed rotation matrix according to (26) we have

$$\begin{aligned}\mathbf{C}(\overline{\boldsymbol\theta}+\delta\boldsymbol\theta)\mathbf{C}(\overline{\boldsymbol\theta}+\delta\boldsymbol\theta)^T &= ((\mathbf{1}-\delta\boldsymbol\phi^{\times})\mathbf{C}(\overline{\boldsymbol\theta}))((\mathbf{1}-\delta\boldsymbol\phi^{\times})\mathbf{C}(\overline{\boldsymbol\theta}))^T \\ &= \mathbf{1} - \delta\boldsymbol\phi^{\times}\delta\boldsymbol\phi^{\times},\end{aligned} \tag{27}$$

which we see is correct to first order in $\delta\boldsymbol\phi$. For this reason, this approach to linearization may be thought of as constraint-sensitive.

Working in the other direction, suppose we have a perturbation in the form of a rotation vector, $\delta\boldsymbol\phi$, and we wish to apply this to a prior value of the rotation, $\mathbf{C}(\overline{\boldsymbol\theta})$. In terms of Euler angles, we would like to carry out the update

$$\boldsymbol\theta = \overline{\boldsymbol\theta} + \mathbf{S}(\overline{\boldsymbol\theta})^{-1}\delta\boldsymbol\phi. \tag{28}$$

However, we would prefer not to use the Euler angles, because $\mathbf{S}(\overline{\boldsymbol\theta})^{-1}$ does not exist precisely at the associated singularities. Instead, we would like to simply store and update the rotation as a rotation matrix. The updated rotation matrix, corresponding to the updated Euler angle sequence above, is given by

$$\mathbf{C}(\boldsymbol\theta) = \mathbf{C}\left(\overline{\boldsymbol\theta} + \mathbf{S}(\overline{\boldsymbol\theta})^{-1}\delta\boldsymbol\phi\right) \tag{29a}$$

$$\approx \left(\mathbf{1} - (\underbrace{\mathbf{S}(\overline{\boldsymbol\theta})\mathbf{S}(\overline{\boldsymbol\theta})^{-1}}_{\mathbf{1}}\delta\boldsymbol\phi)^{\times}\right)\mathbf{C}(\overline{\boldsymbol\theta}) \tag{29b}$$

$$\approx (\mathbf{1} - \delta\boldsymbol\phi^{\times})\mathbf{C}(\overline{\boldsymbol\theta}), \tag{29c}$$

where we have used (26), our linearized rotation matrix expression. We then make the observation that setting $\overline{\boldsymbol\theta} = \mathbf{0}$ in this last expression reveals

$$\mathbf{C}(\delta\boldsymbol\phi) = \mathbf{C}\left(\mathbf{0} + \underbrace{\mathbf{S}(\mathbf{0})^{-1}}_{\mathbf{1}}\delta\boldsymbol\phi\right) \tag{30a}$$

$$\approx (\mathbf{1} - \delta\boldsymbol\phi^{\times})\underbrace{\mathbf{C}(\mathbf{0})}_{\mathbf{1}} \tag{30b}$$

$$\approx (\mathbf{1} - \delta\boldsymbol\phi^{\times}). \tag{30c}$$

Using $\delta\boldsymbol\phi$ as an Euler angle sequence to construct a rotation matrix, $\mathbf{C}(\delta\boldsymbol\phi)$, is somewhat unsettling (since $\delta\boldsymbol\phi$ are not Euler angles), but in the neighbourhood of $\overline{\boldsymbol\theta} = \mathbf{0}$, $\delta\boldsymbol\phi \approx \delta\boldsymbol\theta$, so this is reasonable. In fact, *any* Euler sequence could be used to compute $\mathbf{C}(\delta\boldsymbol\phi)$, as they all result in the same linearized expression. Substituting (30c) into (29c), we arrive at an expression for our rotation matrix update,

$$\mathbf{C}(\boldsymbol\theta) = \mathbf{C}(\delta\boldsymbol\phi)\mathbf{C}(\overline{\boldsymbol\theta}), \tag{31}$$

where we have dropped the approximation symbol due to the fact that the rotation matrix constraint, $\mathbf{C}(\boldsymbol\theta)\mathbf{C}(\boldsymbol\theta)^T = \mathbf{1}$, is satisfied. This update approach allows us to store and update the rotation as a rotation matrix, thereby avoiding singularities and the need to restore the constraint afterwards (i.e., constraint restoration is built in).

3.2. Unit-length quaternion approach

A similar approach may be followed in the treatment of unit-length quaternions. In terms of the axis-angle rotation variables, $\mathbf{a}$ and φ, a unit-length quaternion may be written as

$$\mathbf{q}(\mathbf{a},\varphi) = \begin{bmatrix} \sin\frac{\varphi}{2}\mathbf{a} \\ \cos\frac{\varphi}{2} \end{bmatrix}. \tag{32}$$

Proceeding in a similar manner to the rotation matrix case, we see that

$$\frac{\partial\mathbf{q}(\mathbf{a},\varphi)}{\partial\varphi} = \begin{bmatrix} \frac{1}{2}\cos\frac{\varphi}{2}\mathbf{a} \\ -\frac{1}{2}\sin\frac{\varphi}{2} \end{bmatrix} = \frac{1}{2}\begin{bmatrix} -\mathbf{a}^{\times} & \mathbf{a} \\ -\mathbf{a}^T & 0 \end{bmatrix}\begin{bmatrix} \sin\frac{\varphi}{2}\mathbf{a} \\ \cos\frac{\varphi}{2} \end{bmatrix}, \tag{33}$$

whereupon we have the first key identity,

$$\frac{\partial\mathbf{q}(\mathbf{a},\varphi)}{\partial\varphi} \equiv \frac{1}{2}\begin{bmatrix} \mathbf{a} \\ 0 \end{bmatrix}^{+}\mathbf{q}(\mathbf{a},\varphi), \tag{34}$$

which may be compared to (16) for the rotation matrix case. A principal-axis rotation, $\mathbf{q}_{\psi}(\theta)$, about principal axis ψ and through angle θ, may be written as

$$\mathbf{q}_{\psi}(\theta) := \begin{bmatrix} \sin\frac{\theta}{2}\mathbf{1}_{\psi} \\ \cos\frac{\theta}{2} \end{bmatrix}, \tag{35}$$

where $\mathbf{1}_\psi$ is column ψ of the 3×3 identity matrix [22]. We then have

$$\frac{\partial \mathbf{q}_\psi(\theta)}{\partial \theta} = \frac{1}{2}\begin{bmatrix}\mathbf{1}_\psi \\ 0\end{bmatrix}^+ \mathbf{q}_\psi(\theta). \tag{36}$$

We again consider an α–β–γ Euler sequence (with $\alpha \neq \beta$ and $\beta \neq \gamma$),

$$\mathbf{q}(\boldsymbol{\theta}) := \mathbf{q}_\gamma(\theta_3)^+ \mathbf{q}_\beta(\theta_2)^+ \mathbf{q}_\alpha(\theta_1), \tag{37}$$

where $\boldsymbol{\theta} := [\theta_1 \;\; \theta_2 \;\; \theta_3]^T$. The partial derivatives with respect to the Euler angles are given by

$$\frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \theta_3} = \begin{bmatrix}\frac{1}{2}\mathbf{1}_\gamma \\ 0\end{bmatrix}^+ \mathbf{q}_\gamma(\theta_3)^+ \mathbf{q}_\beta(\theta_2)^+ \mathbf{q}_\alpha(\theta_1) = \frac{1}{2}\mathbf{q}(\boldsymbol{\theta})^\oplus \begin{bmatrix}\mathbf{1}_\gamma \\ 0\end{bmatrix}, \tag{38a}$$

$$\frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \theta_2} = \mathbf{q}_\gamma(\theta_3)^+ \begin{bmatrix}\frac{1}{2}\mathbf{1}_\beta \\ 0\end{bmatrix}^+ \mathbf{q}_\beta(\theta_2)^+ \mathbf{q}_\alpha(\theta_1) = \frac{1}{2}\mathbf{q}(\boldsymbol{\theta})^\oplus \begin{bmatrix}\mathbf{C}_\gamma(\theta_3)\mathbf{1}_\beta \\ 0\end{bmatrix}, \tag{38b}$$

$$\begin{aligned}\frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \theta_1} &= \mathbf{q}_\gamma(\theta_3)^+ \mathbf{q}_\beta(\theta_2)^+ \begin{bmatrix}\frac{1}{2}\mathbf{1}_\alpha \\ 0\end{bmatrix}^+ \mathbf{q}_\alpha(\theta_1) \\ &= \frac{1}{2}\mathbf{q}(\boldsymbol{\theta})^\oplus \begin{bmatrix}\mathbf{C}_\gamma(\theta_3)\mathbf{C}_\beta(\theta_2)\mathbf{1}_\alpha \\ 0\end{bmatrix},\end{aligned} \tag{38c}$$

where we have employed several of our quaternion identities to arrive at the expressions on the far right. Combining the partial derivatives we have

$$\begin{aligned}\frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= \begin{bmatrix}\frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \theta_1} & \frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \theta_2} & \frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \theta_3}\end{bmatrix} \\ &= \frac{1}{2}\mathbf{q}(\boldsymbol{\theta})^\oplus \underbrace{\begin{bmatrix}\mathbf{C}_\gamma(\theta_3)\mathbf{C}_\beta(\theta_2)\mathbf{1}_\alpha & \mathbf{C}_\gamma(\theta_3)\mathbf{1}_\beta & \mathbf{1}_\gamma \\ 0 & 0 & 0\end{bmatrix}}_{\begin{bmatrix}\mathbf{S}(\boldsymbol{\theta}) \\ \mathbf{0}^T\end{bmatrix}},\end{aligned} \tag{39}$$

which is a 4×3 matrix. Thus, we have the second key identity,

$$\frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \equiv \frac{1}{2}\mathbf{q}(\boldsymbol{\theta})^\oplus \begin{bmatrix}\mathbf{S}(\boldsymbol{\theta}) \\ \mathbf{0}^T\end{bmatrix}, \tag{40}$$

which may be compared to (22) for the rotation matrix case. It is also worth noting at this point that

$$\mathbf{q}(\boldsymbol{\theta})^+ \mathbf{q}(\boldsymbol{\theta})^{-1} = \boldsymbol{\iota} \;\Rightarrow\; \mathbf{q}(\boldsymbol{\theta})^+ \frac{\partial \mathbf{q}(\boldsymbol{\theta})^{-1}}{\partial \boldsymbol{\theta}} + \mathbf{q}(\boldsymbol{\theta})^{-1\oplus} \frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0}, \tag{41}$$

so that combining with (40) we have

$$\frac{\partial \mathbf{q}(\boldsymbol{\theta})^{-1}}{\partial \boldsymbol{\theta}} \equiv -\frac{1}{2}\mathbf{q}(\boldsymbol{\theta})^{-1+} \begin{bmatrix}\mathbf{S}(\boldsymbol{\theta}) \\ \mathbf{0}^T\end{bmatrix}. \tag{42}$$

As in the rotation matrix case, we consider perturbing $\mathbf{q}(\boldsymbol{\theta})$ with respect to Euler angles $\boldsymbol{\theta}$. Applying a first-order Taylor approximation we have

$$\mathbf{q}(\bar{\boldsymbol{\theta}} + \delta\boldsymbol{\theta}) \approx \mathbf{q}(\bar{\boldsymbol{\theta}}) + \left.\frac{\partial \mathbf{q}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\bar{\boldsymbol{\theta}}} \delta\boldsymbol{\theta} \tag{43a}$$

$$= \mathbf{q}(\bar{\boldsymbol{\theta}}) + \frac{1}{2}\mathbf{q}(\boldsymbol{\theta})^\oplus \left.\begin{bmatrix}\mathbf{S}(\boldsymbol{\theta}) \\ \mathbf{0}^T\end{bmatrix}\right|_{\bar{\boldsymbol{\theta}}} \delta\boldsymbol{\theta} \tag{43b}$$

$$= \mathbf{q}(\bar{\boldsymbol{\theta}}) + \frac{1}{2}\mathbf{q}(\bar{\boldsymbol{\theta}})^\oplus \begin{bmatrix}\mathbf{S}(\boldsymbol{\theta})\delta\boldsymbol{\theta} \\ 0\end{bmatrix} \tag{43c}$$

$$= \boldsymbol{\iota}^+ \mathbf{q}(\bar{\boldsymbol{\theta}}) + \begin{bmatrix}\frac{1}{2}\mathbf{S}(\bar{\boldsymbol{\theta}})\delta\boldsymbol{\theta} \\ 0\end{bmatrix}^+ \mathbf{q}(\bar{\boldsymbol{\theta}}). \tag{43d}$$

Factoring $\mathbf{q}(\bar{\boldsymbol{\theta}})$ out on the right we arrive at

$$\mathbf{q}(\bar{\boldsymbol{\theta}} + \delta\boldsymbol{\theta}) \approx \underbrace{\left(\boldsymbol{\iota} + \begin{bmatrix}\frac{1}{2}\mathbf{S}(\bar{\boldsymbol{\theta}})\delta\boldsymbol{\theta} \\ 0\end{bmatrix}\right)^+}_{\text{infinitesimal rot.}} \mathbf{q}(\bar{\boldsymbol{\theta}}), \tag{44}$$

which is our expression for perturbing a unit-length quaternion and may be compared to (26) from the rotation matrix case. Making the substitution $\delta\boldsymbol{\phi} := \mathbf{S}(\bar{\boldsymbol{\theta}})\delta\boldsymbol{\theta}$, we have

$$\mathbf{q}(\bar{\boldsymbol{\theta}} + \delta\boldsymbol{\theta}) \approx \left(\boldsymbol{\iota} + \begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi} \\ 0\end{bmatrix}\right)^+ \mathbf{q}(\bar{\boldsymbol{\theta}}) = \underbrace{\begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi} \\ 1\end{bmatrix}}_{\delta\mathbf{q}}^+ \mathbf{q}(\bar{\boldsymbol{\theta}}). \tag{45}$$

This multiplicative perturbation quaternion (it is not unit-length), $\delta\mathbf{q}$, is of the infinitesimal form reported by others [22,15,23]. However, the fact that the perturbation resembles the multiplication of unit-length quaternions (i.e., compounding rotations) is a consequence of our linearization strategy rather than an a priori assumption.

For the unit-length constraint, suppose $\mathbf{q}(\bar{\boldsymbol{\theta}})^T\mathbf{q}(\bar{\boldsymbol{\theta}}) = 1$. Then under the perturbation in (44) we have

$$\begin{aligned}&\mathbf{q}(\bar{\boldsymbol{\theta}} + \delta\boldsymbol{\theta})^T \mathbf{q}(\bar{\boldsymbol{\theta}} + \delta\boldsymbol{\theta}) \\ &= \left(\left(\boldsymbol{\iota} + \begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi} \\ 0\end{bmatrix}\right)^+ \mathbf{q}(\bar{\boldsymbol{\theta}})\right)^T \left(\left(\boldsymbol{\iota} + \begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi} \\ 0\end{bmatrix}\right)^+ \mathbf{q}(\bar{\boldsymbol{\theta}})\right) \\ &= 1 + \mathbf{q}(\bar{\boldsymbol{\theta}})^T \begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi} \\ 0\end{bmatrix}^{+^T} \begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi} \\ 0\end{bmatrix}^+ \mathbf{q}(\bar{\boldsymbol{\theta}}) = 1 + \frac{1}{4}\delta\boldsymbol{\phi}^T\delta\boldsymbol{\phi},\end{aligned} \tag{46}$$

which may be compared to (27). Again, we see that the constraint is satisfied to first order in $\delta\boldsymbol{\phi}$, and thus the linearization approach is constraint-sensitive.

Note that $\mathbf{q}(\bar{\boldsymbol{\theta}} + \delta\boldsymbol{\theta})^T\mathbf{q}(\bar{\boldsymbol{\theta}} + \delta\boldsymbol{\theta})$ is simply the length of $\delta\mathbf{q}$. It is therefore tempting to restore the unit-length constraint on $\mathbf{q}(\bar{\boldsymbol{\theta}} + \delta\boldsymbol{\theta})$ by normalizing $\delta\mathbf{q}$ by its length. However, we can avoid normalization altogether using the approach outlined in the rotation matrix case. Suppose we have a prior value for the rotation, $\mathbf{q}(\bar{\boldsymbol{\theta}})$, and wish to apply a perturbation, $\delta\boldsymbol{\phi}$. The updated unit-length quaternion, corresponding to the updated Euler angle sequence in (28), is given by

$$\mathbf{q}(\boldsymbol{\theta}) = \mathbf{q}\left(\bar{\boldsymbol{\theta}} + \mathbf{S}(\bar{\boldsymbol{\theta}})^{-1}\delta\boldsymbol{\phi}\right) \tag{47a}$$

$$\approx \left(\boldsymbol{\iota} + \begin{bmatrix}\frac{1}{2}\overbrace{\mathbf{S}(\bar{\boldsymbol{\theta}})\mathbf{S}(\bar{\boldsymbol{\theta}})^{-1}}^{\mathbf{1}}\delta\boldsymbol{\phi} \\ 0\end{bmatrix}\right)^+ \mathbf{q}(\bar{\boldsymbol{\theta}}) \tag{47b}$$

$$\approx \left(\boldsymbol{\iota}+\begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi}\\0\end{bmatrix}\right)^{+}\mathbf{q}(\overline{\boldsymbol{\theta}}), \tag{47c}$$

where we have used (44), our linearized rotation matrix expression. We then make the observation that setting $\overline{\boldsymbol{\theta}}=\mathbf{0}$ in this last expression reveals

$$\mathbf{q}(\delta\boldsymbol{\phi})=\mathbf{q}\Big(\mathbf{0}+\underbrace{\mathbf{S}(\mathbf{0})^{-1}}_{\mathbf{1}}\delta\boldsymbol{\phi}\Big) \tag{48a}$$

$$\approx \left(\boldsymbol{\iota}+\begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi}\\0\end{bmatrix}\right)^{+}\underbrace{\mathbf{q}(\mathbf{0})}_{\boldsymbol{\iota}} \tag{48b}$$

$$\approx \left(\boldsymbol{\iota}+\begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi}\\0\end{bmatrix}\right). \tag{48c}$$

As in the rotation matrix case, *any* Euler sequence could be used to compute $\mathbf{q}(\delta\boldsymbol{\phi})$, as they all result in the same linearized expression. Substituting (48c) into (47c), we arrive at an expression for our unit-length quaternion update,

$$\mathbf{q}(\boldsymbol{\theta})=\mathbf{q}(\delta\boldsymbol{\phi})^{+}\mathbf{q}(\overline{\boldsymbol{\theta}}), \tag{49}$$

where we have dropped the approximation symbol due to the fact that the unit-length quaternion constraint, $\mathbf{q}(\boldsymbol{\theta})^T\mathbf{q}(\boldsymbol{\theta})=1$, is satisfied. This update approach allows us to store and update the rotation as a unit-length quaternion, thereby avoiding singularities and the need to restore the constraint afterwards (i.e., constraint restoration is built in).

3.3. *Sun sensor unit-length quaternion example*

In this section we provide an example of using our linearized unit-length quaternion expression to construct a linearized error term for a sun sensor measurement for use in a batch-style attitude estimator. We begin by defining the reference frames depicted in Fig. 1. Suppose the estimation problem is to determine $\mathbf{q}_{si}$, the rotation from the inertial frame, $\underrightarrow{\mathcal{F}}_i$, to the sun sensor frame, $\underrightarrow{\mathcal{F}}_s$. The sun sensor measures the direction to the sun, $\underrightarrow{s}$, in frame $\underrightarrow{\mathcal{F}}_s$; this is a two-degree-of-freedom measurement. In our formulation, we parameterize this type of measurement, $\tilde{\mathbf{y}}$, as

$$\tilde{\mathbf{y}} := \begin{bmatrix}\tilde{\mu}\\\tilde{\lambda}\end{bmatrix}, \tag{50}$$

where $\tilde{\mu}$ and $\tilde{\lambda}$ are noise-corrupted measurements of the true angles μ and λ in Fig. 1. We note that the rotation from $\underrightarrow{\mathcal{F}}_m$ to $\underrightarrow{\mathcal{F}}_s$, $\mathbf{q}_{sm}$, can be written as the Euler sequence

$$\mathbf{q}_{sm}=\mathbf{q}_1(\lambda)^{+}\mathbf{q}_2(\mu)^{+}\mathbf{q}_3(0), \tag{51}$$

where

$$\mathbf{q}_1(\theta):=\begin{bmatrix}\sin\frac{\theta}{2}\mathbf{1}_1\\\cos\frac{\theta}{2}\end{bmatrix},\quad \mathbf{q}_2(\theta):=\begin{bmatrix}\sin\frac{\theta}{2}\mathbf{1}_2\\\cos\frac{\theta}{2}\end{bmatrix},\quad \mathbf{q}_3(\theta):=\begin{bmatrix}\sin\frac{\theta}{2}\mathbf{1}_3\\\cos\frac{\theta}{2}\end{bmatrix}. \tag{52}$$

We also note that the rotation from $\underrightarrow{\mathcal{F}}_e$ to $\underrightarrow{\mathcal{F}}_i$, $\mathbf{q}_{ie}$, can be computed from ephemeris and date/time, and is therefore a known quantity. Finally, the rotation from $\underrightarrow{\mathcal{F}}_m$ to $\underrightarrow{\mathcal{F}}_e$, $\mathbf{q}_{em}$, is a rotation about the z-axis (of either frame), through an unknown angle, ψ. In other words, $\mathbf{q}_{em}=\mathbf{q}_3(\psi)$; this is because the z-axes of both frames point directly at the sun.

With these preliminaries established, we turn to building the sun sensor error term. Noting that $\mathbf{q}_{sm}$ contains the measurement information, we build a predicted version of this, $\hat{\mathbf{q}}_{sm}$, based on our attitude estimate, $\hat{\mathbf{q}}_{si}$, and the other inter-frame rotations:

$$\hat{\mathbf{q}}_{sm}:=\hat{\mathbf{q}}_{si}^{+}\mathbf{q}_{ie}^{+}\mathbf{q}_{em}=\mathbf{q}_1(\hat{\lambda})^{+}\mathbf{q}_2(\hat{\mu})^{+}\mathbf{q}_3(\hat{\nu}), \tag{53}$$

where we have also written it as an Euler sequence on the right. The angle $\hat{\nu}$ is non-zero because we are using $\hat{\mathbf{q}}_{si}$, not the true value, $\mathbf{q}_{si}$. We then perturb the expression for $\hat{\mathbf{q}}_{si}$ about some initial estimate, $\overline{\mathbf{q}}_{si}$, using (44):

$$\hat{\mathbf{q}}_{sm}\approx\underbrace{\left(\boldsymbol{\iota}+\begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\phi}\\0\end{bmatrix}\right)^{+}\overline{\mathbf{q}}_{si}^{+}}_{\approx\hat{\mathbf{q}}_{si}}\mathbf{q}_{ie}^{+}\mathbf{q}_{em}. \tag{54}$$

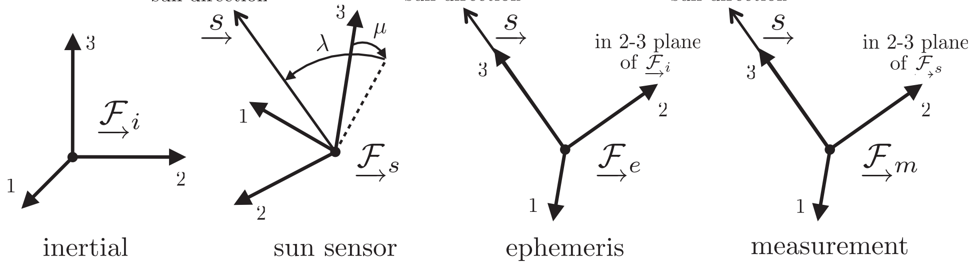

Fig. 1. Reference frames used in sun sensor example.

The compound rotation, $\overline{\mathbf{q}}_{si}^{+}\mathbf{q}_{ie}$, may be expressed as the Euler sequence,

$$\overline{\mathbf{q}}_{si}^{+}\mathbf{q}_{ie} = \mathbf{q}_1(\overline{\lambda})^{+}\mathbf{q}_2(\overline{\mu})^{+}\mathbf{q}_3(\overline{\nu}), \tag{55}$$

so that (54) may be written as

$$\hat{\mathbf{q}}_{sm} \approx \left(\boldsymbol{\iota} + \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\phi} \\ 0 \end{bmatrix}\right)^{+} \mathbf{q}_1(\overline{\lambda})^{+}\mathbf{q}_2(\overline{\mu})^{+}\mathbf{q}_3(\overline{\nu})^{+}\mathbf{q}_3(\psi) \tag{56a}$$

$$\approx \left(\boldsymbol{\iota} + \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\phi} \\ 0 \end{bmatrix}\right)^{+} \mathbf{q}_1(\overline{\lambda})^{+}\mathbf{q}_2(\overline{\mu})^{+}\mathbf{q}_3(\overline{\nu}+\psi) \tag{56b}$$

$$\approx \left(\boldsymbol{\iota} + \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\phi} \\ 0 \end{bmatrix}\right)^{+} \overline{\mathbf{q}}_{sm}, \tag{56c}$$

where $\overline{\mathbf{q}}_{sm} := \mathbf{q}_1(\overline{\lambda})^{+}\mathbf{q}_2(\overline{\mu})^{+}\mathbf{q}_3(\overline{\nu}+\psi)$. The angle ψ is unknown, but, as we will see shortly, we do not need it.

This last expression is precisely of the linearized form in (44), but now for $\hat{\mathbf{q}}_{sm}$ not $\hat{\mathbf{q}}_{si}$. This allows us to immediately rewrite it in terms of the constituent Euler angles,

$$\hat{\boldsymbol{\theta}} \approx \overline{\boldsymbol{\theta}} + \mathbf{S}(\overline{\boldsymbol{\theta}})^{-1}\delta\boldsymbol{\phi}, \tag{57}$$

where

$$\hat{\boldsymbol{\theta}} := \begin{bmatrix} \hat{\nu} \\ \hat{\mu} \\ \hat{\lambda} \end{bmatrix}, \quad \overline{\boldsymbol{\theta}} := \begin{bmatrix} \overline{\nu}+\psi \\ \overline{\mu} \\ \overline{\lambda} \end{bmatrix},$$

$$\mathbf{S}(\overline{\boldsymbol{\theta}}) := \left[\mathbf{C}_1(\overline{\lambda})\mathbf{C}_2(\overline{\mu})\mathbf{1}_3 \;\; \mathbf{C}_1(\overline{\lambda})\mathbf{1}_2 \;\; \mathbf{1}_1\right]. \tag{58}$$

Eq. (57) tells us how the Euler sequence for $\hat{\mathbf{q}}_{sm}$ changes as we perturb it with $\delta\boldsymbol{\phi}$ (in the neighbourhood around the prior estimate, $\overline{\mathbf{q}}_{sm}$). It is important to note that the inverse of $\mathbf{S}(\overline{\boldsymbol{\theta}})$ will always exist for sun sensors with a half-angle field of view less than $\pi/2$ (i.e., a hemisphere). This is because we have chosen an Euler sequence to parameterize $\mathbf{q}_{sm}$ with the singularity at $\mu=\pi/2$. It is also worth noting that the unknown angle, ψ, is not used in the calculation of $\mathbf{S}(\overline{\boldsymbol{\theta}})$, again by careful selection of the Euler sequence.

We can build an expected measurement, $\hat{\mathbf{y}}$, by applying a projection matrix, $\mathbf{P}$, to $\hat{\boldsymbol{\theta}}$:

$$\hat{\mathbf{y}} := \mathbf{P}\hat{\boldsymbol{\theta}} = \begin{bmatrix} \hat{\mu} \\ \hat{\lambda} \end{bmatrix} = \overline{\mathbf{y}} + \mathbf{P}\mathbf{S}(\overline{\boldsymbol{\theta}})^{-1}\delta\boldsymbol{\phi}, \tag{59}$$

where

$$\overline{\mathbf{y}} := \begin{bmatrix} \overline{\mu} \\ \overline{\lambda} \end{bmatrix}, \quad \mathbf{P} := \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{60}$$

The linearized error term, $\mathbf{e}$, for our sun sensor measurement is then simply the difference between the actual measurement and the expected measurement:

$$\mathbf{e} := \tilde{\mathbf{y}} - \overline{\mathbf{y}} - \mathbf{P}\mathbf{S}(\overline{\boldsymbol{\theta}})^{-1}\delta\boldsymbol{\phi}, \tag{61}$$

which is linear in its dependence on $\delta\boldsymbol{\phi}$.

Note that one sun sensor measurement is not enough to solve for attitude alone. However, this error term could be used in a batch estimator along with measurements from other sources such as a stereo camera, inclinometers, star tracker, or angular rate sensors. One iteration of a batch-style estimator would proceed as follows:

1. Begin with an initial attitude estimate, $\overline{\mathbf{q}}_{si}$.
2. Decompose $\overline{\mathbf{q}}_{si}^{+}\mathbf{q}_{ie}$ into Euler angles $\overline{\lambda}$, $\overline{\mu}$, and $\overline{\nu}$ using a 3-2-1 sequence. Note that we will not require $\overline{\nu}$.
3. Build $\overline{\mathbf{y}}$ and $\mathbf{S}(\overline{\boldsymbol{\theta}})$ from $\overline{\lambda}$ and $\overline{\mu}$.
4. Build $\mathbf{e}$ from $\overline{\mathbf{y}}$ and $\mathbf{S}(\overline{\boldsymbol{\theta}})$.
5. Use $\mathbf{e}$ in any sum-of-squared-errors objective function for a batch-style estimator (along with other measurements) to solve for $\delta\boldsymbol{\phi}^{\star}$, which will be some optimal perturbation to the initial attitude estimate.
6. Update $\overline{\mathbf{q}}_{si}$ using $\delta\boldsymbol{\phi}^{\star}$ in (49).
7. Check for convergence. If not converged, return to step 2.

It is worth noting that in this procedure, the attitude we are estimating, $\mathbf{q}_{si}$, is always stored as a unit-length quaternion. We do not need to compute its constituent Euler angles at any time. This implies that the procedure can handle arbitrary motions of the sun sensor frame without encountering singularities. A similar procedure exists for the rotation matrix representation of $\mathbf{q}_{si}$.

4. Weighted-least-squares point cloud alignment

In this section we provide an extended example of a pose estimation problem: weighted-least-squares point-cloud alignment. The problem setup is depicted in Fig. 2. There are two reference frames, $\underrightarrow{\mathcal{F}}_u$ and $\underrightarrow{\mathcal{F}}_v$, which could represent the pose of a moving vehicle or camera at two different times, for example. For the measurements, we have M pairs of vector observations, $(\underrightarrow{u}_j, \underrightarrow{v}_j)$, where $j = 1,\ldots,M$. Each pair is a correlated set of measurements of the same point, P_j, on some stationary object. We assume all measurements are corrupted by some noise. Our task is to estimate the pose (i.e., translation and rotation) from $\underrightarrow{\mathcal{F}}_u$ to $\underrightarrow{\mathcal{F}}_v$, based on the measurements. We assume that we do not know the true position of the points, P_j, in either frame. This situation is relevant to spacecraft rendezvous and docking as well as visual motion estimation of a planetary rover.

There are two approaches we may take to estimate pose: *scalar weights* and *matrix weights*. In the scalar-weight case, we assume the noise corrupting each measurement is isotropic (i.e., the same in all directions

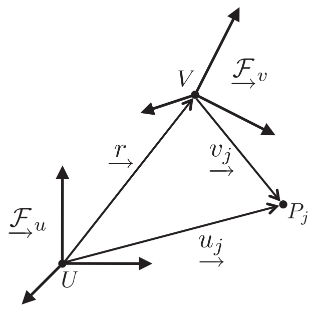

Fig. 2. References frames for point cloud alignment.

about P_j) and therefore we can minimize an objective function that uses simple scalar weights. In this case, we can solve for an optimal estimate of the pose closed form. In the matrix-weight case, we assume the nose is anisotropic (i.e., different in different directions) and therefore require an objective function with matrix weights. In this case, the pose problem must be solved iteratively using, for example, a Gauss–Newton algorithm [20]. The latter iterative scheme requires an initial guess, which may be obtained by first solving the scalar-weight problem. We present both solutions, treating the scalar-weight case with unit-length quaternions and the matrix-weight case with rotation matrices.

4.1. Scalar weights

The scalar-weight case can be solved in closed form [8]. We show this result in our notation for comparison to the next section on the matrix-weight case. We will use quaternions in this case and thus we define the following 4×1 columns:

$\mathbf{q}$: (unit – length) rotation from $\underrightarrow{\mathcal{F}}_u$ to $\underrightarrow{\mathcal{F}}_v$;

$\mathbf{r}$: translation of V with respect to U, expressed in $\underrightarrow{\mathcal{F}}_u$;

$\mathbf{u}_j$: measurement of P_j with respect to U, expressed in $\underrightarrow{\mathcal{F}}_u$;

$\mathbf{v}_j$: measurement of P_j with respect to V, expressed in $\underrightarrow{\mathcal{F}}_v$.

In the absence of any noise, the geometry of the situation is described by

$$\mathbf{v}_j = \mathbf{q}^+(\mathbf{u}_j - \mathbf{r})^+\mathbf{q}^{-1}. \tag{62}$$

Referring to (62), we could define an error quaternion for the measurements associated with point P_j according to

$$\hat{\mathbf{e}}_j := \mathbf{v}_j - \mathbf{q}^+(\mathbf{u}_j - \mathbf{r})^+\mathbf{q}^{-1}, \tag{63}$$

but instead we can manipulate the above to generate an error that appears linear in $\mathbf{q}$,

$$\mathbf{e}_j := \mathbf{q}^{\oplus}\hat{\mathbf{e}}_j = (\mathbf{v}_j^+ - (\mathbf{u}_j - \mathbf{r})^{\oplus})\mathbf{q}. \tag{64}$$

We will define the total sum-of-squares objective function (to minimize), J, as

$$J(\mathbf{q},\mathbf{r},\lambda) := \frac{1}{2}\sum_{j=1}^{M} w_j \mathbf{e}_j^T\mathbf{e}_j - \underbrace{\frac{1}{2}\lambda(\mathbf{q}^T\mathbf{q}-1)}_{\text{Lagrange mult. term}}, \tag{65}$$

where the w_j are unique scalar weights assigned to each of the point pairs. This is similar to the classic Wahba's problem [30], but we also have a translation to estimate. We have included the Lagrange multiplier term on the right to enforce the unit-length constraint on the rotation quaternion. It is also worth noting that selecting $\mathbf{e}_j$ over $\hat{\mathbf{e}}_j$ has no effect on our objective function since

$$\mathbf{e}_j^T\mathbf{e}_j = (\mathbf{q}^{\oplus}\hat{\mathbf{e}}_j)^T(\mathbf{q}^{\oplus}\hat{\mathbf{e}}_j) = \hat{\mathbf{e}}_j^T\mathbf{q}^{\oplus^T}\mathbf{q}^{\oplus}\hat{\mathbf{e}}_j = \hat{\mathbf{e}}_j^T(\mathbf{q}^{-1^{\oplus}}\mathbf{q})^{\oplus}\hat{\mathbf{e}}_j = \hat{\mathbf{e}}_j^T\hat{\mathbf{e}}_j. \tag{66}$$

Inserting the expression for $\mathbf{e}_j$ into the objective function we see

$$J(\mathbf{q},\mathbf{r},\lambda) = \frac{1}{2}\sum_{j=1}^{M} w_j\mathbf{q}^T(\mathbf{v}_j^+ - (\mathbf{u}_j - \mathbf{r})^{\oplus})^T(\mathbf{v}_j^+ - (\mathbf{u}_j - \mathbf{r})^{\oplus})\mathbf{q} - \frac{1}{2}\lambda(\mathbf{q}^T\mathbf{q}-1). \tag{67}$$

Taking the derivative of the objective function with respect to $\mathbf{q}$, $\mathbf{r}$, and λ we find

$$\frac{\partial J}{\partial \mathbf{q}}^T = \sum_{j=1}^{M} w_j(\mathbf{v}_j^+ - (\mathbf{u}_j - \mathbf{r})^{\oplus})^T(\mathbf{v}_j^+ - (\mathbf{u}_j - \mathbf{r})^{\oplus})\mathbf{q} - \lambda\mathbf{q}, \tag{68a}$$

$$\frac{\partial J}{\partial \mathbf{r}}^T = \mathbf{q}^{-1^+}\sum_{j=1}^{M} w_j(\mathbf{v}_j^+ - (\mathbf{u}_j - \mathbf{r})^{\oplus})\mathbf{q}, \tag{68b}$$

$$\frac{\partial J}{\partial \lambda} = -\frac{1}{2}(\mathbf{q}^T\mathbf{q}-1). \tag{68c}$$

Setting the second to zero we find

$$\mathbf{r} = \mathbf{u} - \mathbf{q}^{-1^+}\mathbf{v}^+\mathbf{q}, \tag{69}$$

where $\mathbf{u}$ and $\mathbf{v}$ are defined below. Substituting $\mathbf{r}$ into the first and setting to zero we can show

$$\mathbf{A}\mathbf{q} = \lambda\mathbf{q}, \tag{70}$$

where

$$\mathbf{A} := \frac{1}{w}\sum_{j=1}^{M} w_j((\mathbf{v}_j - \mathbf{v})^+ - (\mathbf{u}_j - \mathbf{u})^{\oplus})^T((\mathbf{v}_j - \mathbf{v})^+ - (\mathbf{u}_j - \mathbf{u})^{\oplus}), \tag{71a}$$

$$\mathbf{v} := \frac{1}{w}\sum_{j=1}^{M} w_j\mathbf{v}_j, \quad \mathbf{u} := \frac{1}{w}\sum_{j=1}^{M} w_j\mathbf{u}_j, \quad w := \sum_{j=1}^{M} w_j. \tag{71b}$$

We can see this is just an eigenproblem. Finding the smallest eigenvalue and the corresponding eigenvector will yield $\mathbf{q}$ to within a multiplicative constant and our constraint that $\mathbf{q}^T\mathbf{q}=1$ makes the solution unique.

To see that we want the smallest eigenvalue, we return to the objective function. We can see from setting (68a) to zero that an equivalent expression for $\mathbf{A}$ is

$$\mathbf{A} = \sum_{j=1}^{M} w_j(\mathbf{v}_j^+ - (\mathbf{u}_j - \mathbf{r})^{\oplus})^T(\mathbf{v}_j^+ - (\mathbf{u}_j - \mathbf{r})^{\oplus}). \tag{72}$$

Substituting this into the objective function in (67) we immediately see that

$$J(\mathbf{q},\mathbf{r},\lambda) = \frac{1}{2}\mathbf{q}^T\underbrace{\mathbf{A}\mathbf{q}}_{\lambda\mathbf{q}} - \frac{1}{2}\lambda(\mathbf{q}^T\mathbf{q}-1) = \frac{1}{2}\lambda. \tag{73}$$

Thus, picking the smallest possible value for λ will minimize the objective function.

Note, we have not made any approximations or linearizations, but this depends heavily on the fact that the weights are scalar not matrices.

4.2. Matrix weights

In some situations, the assumption of isotropic measurement noise is not appropriate. For example, if the measurements of P_j are obtained from a stereo camera, there will be much more noise in the depth direction than the lateral directions [18]. In this case, we can employ a sum-of-squares objective function of the form $J := \frac{1}{2}\sum_{j=1}^{M} \mathbf{e}_j^T \mathbf{W}_j \mathbf{e}_j$, where $\mathbf{e}_j$ is a measurement error term and $\mathbf{W}_j$ is the associated weight. A common approach for the selection of $\mathbf{W}_j$ is to use the inverse covariance matrix of $\mathbf{e}_j$. In this case, J takes on the form of a Mahalanobis distance [13]. If we select $\mathbf{e}_j$ to be of the form in (64), we find that the covariance matrix, and hence $\mathbf{W}_j$, depends on the state we are trying to estimate [19,14], which complicates the solution.

Instead, we turn to a well-known approach in the robotics and computer vision domain. Namely, we introduce the positions of the points, P_j, as part of the state we are trying to estimate. This augmented state estimation problem has been called *simultaneous localization and mapping* [25,26] and *bundle adjustment* [29,2] in the robotics and computer vision literature. We will see that although the size of the state is much larger, we will be able to exploit the sparse structure of the problem to keep the computational cost low.

We will use rotation matrices in this example and thus define the following quantities:

$\mathbf{C}$: 3×3 rotation matrix from $\underrightarrow{\mathcal{F}}_u$ to $\underrightarrow{\mathcal{F}}_v$;

$\mathbf{r}$: 3×1 translation of V with respect to U, expressed in $\underrightarrow{\mathcal{F}}_u$;

$\mathbf{p}_j$: 3×1 translation of P_j with respect to U, expressed in $\underrightarrow{\mathcal{F}}_u$;

$\mathbf{u}_j$: 3×1 measurement of P_j with respect to U, expressed in $\underrightarrow{\mathcal{F}}_u$;

$\mathbf{v}_j$: 3×1 measurement of P_j with respect to V, expressed in $\underrightarrow{\mathcal{F}}_v$.

In the absence of any noise, the geometry of the situation is described by

$$\mathbf{v}_j = \mathbf{C}(\mathbf{u}_j - \mathbf{r}). \tag{74}$$

With this setup, our objective function and errors are defined as

$$J(\mathbf{C},\mathbf{r},\mathbf{p}_j) := \frac{1}{2}\sum_{j=1}^{M}(\mathbf{e}_{u,j}^T \mathbf{U}_j^{-1}\mathbf{e}_{u,j} + \mathbf{e}_{v,j}^T \mathbf{V}_j^{-1}\mathbf{e}_{v,j}), \tag{75a}$$

$$\mathbf{e}_{u,j} := \mathbf{u}_j - \mathbf{p}_j, \tag{75b}$$

$$\mathbf{e}_{v,j} := \mathbf{v}_j - \mathbf{C}(\mathbf{p}_j - \mathbf{r}), \tag{75c}$$

where $\mathbf{U}_j$ is the covariance of $\mathbf{e}_{u,j}$ and $\mathbf{V}_j$ is the covariance of $\mathbf{e}_{v,j}$. We seek to minimize the objective function using the Gauss–Newton algorithm [20]. It is straightforward to perturb the error terms using

$$\mathbf{r} = \bar{\mathbf{r}} + \delta\mathbf{r}, \tag{76a}$$

$$\mathbf{C} = \delta\mathbf{C}\,\bar{\mathbf{C}} \approx (\mathbf{1} - \delta\boldsymbol{\phi}^\times)\bar{\mathbf{C}}, \tag{76b}$$

$$\mathbf{p}_j = \bar{\mathbf{p}}_j + \delta\mathbf{p}_j, \tag{76c}$$

where $\overline{(\cdot)}$ are the initial guesses and $\delta(\cdot)$ the perturbations. Eq. (76b) employs our linearized rotation matrix expression derived earlier. Substituting these into the error terms we have

$$\mathbf{e}_{u,j} \approx \underbrace{\mathbf{u}_j - \bar{\mathbf{p}}_j}_{\bar{\mathbf{e}}_{u,j}} - \underbrace{\left[\mathbf{0}\ \mathbf{0}\ \middle|\ \overbrace{\mathbf{0}\ \cdots\ \mathbf{1}\ \cdots\ \mathbf{0}}^{\text{only col. } j \text{ is non-zero}}\right]}_{\mathbf{H}_{u,j}} \delta\mathbf{x}, \tag{77a}$$

$$\mathbf{e}_{v,j} \approx \underbrace{\mathbf{v}_j - \bar{\mathbf{C}}(\bar{\mathbf{p}}_j - \bar{\mathbf{r}})}_{\bar{\mathbf{e}}_{v,j}} - \underbrace{\left[-\bar{\mathbf{C}}\ \ (\bar{\mathbf{C}}(\bar{\mathbf{p}}_j - \bar{\mathbf{r}}))^\times\ \middle|\ \overbrace{\mathbf{0}\ \cdots\ \bar{\mathbf{C}}\ \cdots\ \mathbf{0}}^{\text{only col. } j \text{ is non-zero}}\right]}_{\mathbf{H}_{v,j}} \delta\mathbf{x}, \tag{77b}$$

$$\delta\mathbf{x} := \left[\delta\mathbf{r}^T\ \ \delta\boldsymbol{\phi}^T\ \middle|\ \delta\mathbf{p}_1^T\ \cdots\ \delta\mathbf{p}_M^T\right]^T, \tag{77c}$$

where we have dropped products of small terms. These errors now have a linear dependence (with an offset) on $\delta\mathbf{x}$, the perturbations. Note that we have shown the state partitioning between the pose and the positions of the points with vertical lines. Substituting these linearized error expressions in the objective function, J, it becomes quadratic in $\delta\mathbf{x}$. It is straightforward to show that the value of $\delta\mathbf{x}$ that minimizes J is given by $\delta\mathbf{x}^\star$, the solution to the following linear system of equations:

$$\underbrace{\sum_{j=1}^{M}(\mathbf{H}_{u,j}^T\mathbf{U}_j^{-1}\mathbf{H}_{u,j} + \mathbf{H}_{v,j}^T\mathbf{V}_j^{-1}\mathbf{H}_{v,j})}_{\mathbf{A}}\delta\mathbf{x}^\star = \underbrace{\sum_{j=1}^{M}(\mathbf{H}_{u,j}^T\mathbf{U}_j^{-1}\bar{\mathbf{e}}_{u,j} + \mathbf{H}_{v,j}^T\mathbf{V}_j^{-1}\bar{\mathbf{e}}_{v,j})}_{\mathbf{b}}. \tag{78}$$

We can write this in partitioned form as

$$\underbrace{\left[\begin{array}{c|c}\mathbf{A}_{11} & \mathbf{A}_{12} \\ \hline \mathbf{A}_{12}^T & \mathbf{A}_{22}\end{array}\right]}_{\mathbf{A}} \underbrace{\left[\begin{array}{c}\delta\mathbf{x}_1^\star \\ \hline \delta\mathbf{x}_2^\star\end{array}\right]}_{\delta\mathbf{x}^\star} = \underbrace{\left[\begin{array}{c}\mathbf{b}_1 \\ \hline \mathbf{b}_2\end{array}\right]}_{\mathbf{b}}, \tag{79}$$

where $\delta\mathbf{x}_1^\star$ are the pose state variables and $\delta\mathbf{x}_2^\star$ are the point position state variables. In detail we have

$$\mathbf{A}_{11} = \begin{bmatrix} \sum_{j=1}^{M}\bar{\mathbf{C}}^T\mathbf{V}_j^{-1}\bar{\mathbf{C}} & -\sum_{j=1}^{M}\bar{\mathbf{C}}^T\mathbf{V}_j^{-1}(\bar{\mathbf{C}}(\bar{\mathbf{p}}_j-\bar{\mathbf{r}}))^\times \\ \sum_{j=1}^{M}(\bar{\mathbf{C}}(\bar{\mathbf{p}}_j-\bar{\mathbf{r}}))^\times\mathbf{V}_j^{-1}\bar{\mathbf{C}} & -\sum_{j=1}^{M}(\bar{\mathbf{C}}(\bar{\mathbf{p}}_j-\bar{\mathbf{r}}))^\times\mathbf{V}_j^{-1}(\bar{\mathbf{C}}(\bar{\mathbf{p}}_j-\bar{\mathbf{r}}))^\times \end{bmatrix}, \tag{80a}$$

$$\mathbf{A}_{12} = \begin{bmatrix} -\bar{\mathbf{C}}^T\mathbf{V}_1^{-1}\bar{\mathbf{C}} & \cdots & -\bar{\mathbf{C}}^T\mathbf{V}_M^{-1}\bar{\mathbf{C}} \\ -(\bar{\mathbf{C}}(\bar{\mathbf{p}}_1-\bar{\mathbf{r}}))^\times\mathbf{V}_1^{-1}\bar{\mathbf{C}} & \cdots & -(\bar{\mathbf{C}}(\bar{\mathbf{p}}_M-\bar{\mathbf{r}}))^\times\mathbf{V}_M^{-1}\bar{\mathbf{C}} \end{bmatrix}, \tag{80b}$$

$$\mathbf{A}_{22} = \operatorname{diag}\left\{\mathbf{U}_1^{-1} + \bar{\mathbf{C}}^T\mathbf{V}_1^{-1}\bar{\mathbf{C}}, \ldots, \mathbf{U}_M^{-1} + \bar{\mathbf{C}}^T\mathbf{V}_M^{-1}\bar{\mathbf{C}}\right\}, \tag{80c}$$

$$\mathbf{b}_1 = \begin{bmatrix} -\sum_{j=1}^{M}\bar{\mathbf{C}}^T\mathbf{V}_j^{-1}\bar{\mathbf{e}}_{v,j} \\ -\sum_{j=1}^{M}(\bar{\mathbf{C}}(\bar{\mathbf{p}}_j-\bar{\mathbf{r}}))^\times\mathbf{V}_j^{-1}\bar{\mathbf{e}}_{v,j} \end{bmatrix}, \tag{80d}$$

$$\mathbf{b}_2 = \begin{bmatrix} \mathbf{U}_1^{-1}\bar{\mathbf{e}}_{u,1} + \bar{\mathbf{C}}^T\mathbf{V}_1^{-1}\bar{\mathbf{e}}_{v,1} \\ \vdots \\ \mathbf{U}_M^{-1}\bar{\mathbf{e}}_{u,M} + \bar{\mathbf{C}}^T\mathbf{V}_M^{-1}\bar{\mathbf{e}}_{v,M} \end{bmatrix}. \tag{80e}$$

This system may be solved efficiently using the Schur complement due to the fact that $\mathbf{A}_{22}$ is block-diagonal; this is referred to as *sparse bundle adjustment* in the computer vision literature [2,29]. Premultiplying both sides of (79) by

$$\left[\begin{array}{c|c} \mathbf{1} & -\mathbf{A}_{12}\mathbf{A}_{22}^{-1} \\ \hline \mathbf{0} & \mathbf{1} \end{array}\right] \tag{81}$$

results in

$$\left[\begin{array}{c|c} \mathbf{A}_{11}-\mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{12}^T & \mathbf{0} \\ \hline \mathbf{A}_{12}^T & \mathbf{A}_{22} \end{array}\right]\left[\begin{array}{c} \delta\mathbf{x}_1^\star \\ \hline \delta\mathbf{x}_2^\star \end{array}\right] = \left[\begin{array}{c} \mathbf{b}_1-\mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{b}_2 \\ \hline \mathbf{b}_2 \end{array}\right], \tag{82}$$

which has the same solution as the original system. The advantage is that $\mathbf{A}_{22}$ can be inverted efficiently due to its block-diagonal form. The pose state variables, $\delta\mathbf{x}_1^\star$, can be determined directly from (82) and the point position variables, $\delta\mathbf{x}_2^\star$, found inexpensively through back-substitution. After solving for $\delta\mathbf{x}^\star$ we update $\overline{\mathbf{r}}$ using (76a), $\overline{\mathbf{p}}_j$ using (76c), and $\overline{\mathbf{C}}$ using (31). We iterate through the entire procedure until $\delta\mathbf{x}^\star$ is sufficiently small. Once converged, the covariance of the state estimate is provided by $\mathbf{A}^{-1}$. It should be pointed out that this approach does not require us to decompose $\mathbf{C}$ into an Euler angle sequence at any time and thus is entirely general.

4.3. Numerical example

In this section we provide a numerical example of aligning two point clouds using the Gauss–Newton method from the previous section. Fig. 3(a) depicts two sets of noisy point cloud measurements generated by a simulated stereo camera. Due to the nature of this camera, the resulting covariance matrices, $\mathbf{U}_j$ and $\mathbf{V}_j$, are as shown in the plot [18]. Measurements further from the camera (represented by the axes in the plot) are noisier than those close to the camera. It is therefore important to use a matrix weighted-least-squares technique, so that we may trust the measurements closer to the camera more and those further from the camera less. Accordingly, we employed the method from the previous section to estimate $\mathbf{C}$ and $\mathbf{r}$ (as well as $\mathbf{p}_j$). For the initial condition we used $\overline{\mathbf{C}}=\mathbf{1}$, $\overline{\mathbf{r}}=\mathbf{0}$, and $\overline{\mathbf{p}}_j=\overline{\mathbf{C}}^T\mathbf{v}_j+\overline{\mathbf{r}}$. Fig. 3(b) depicts the convergence of the objection function, J, with subsequent

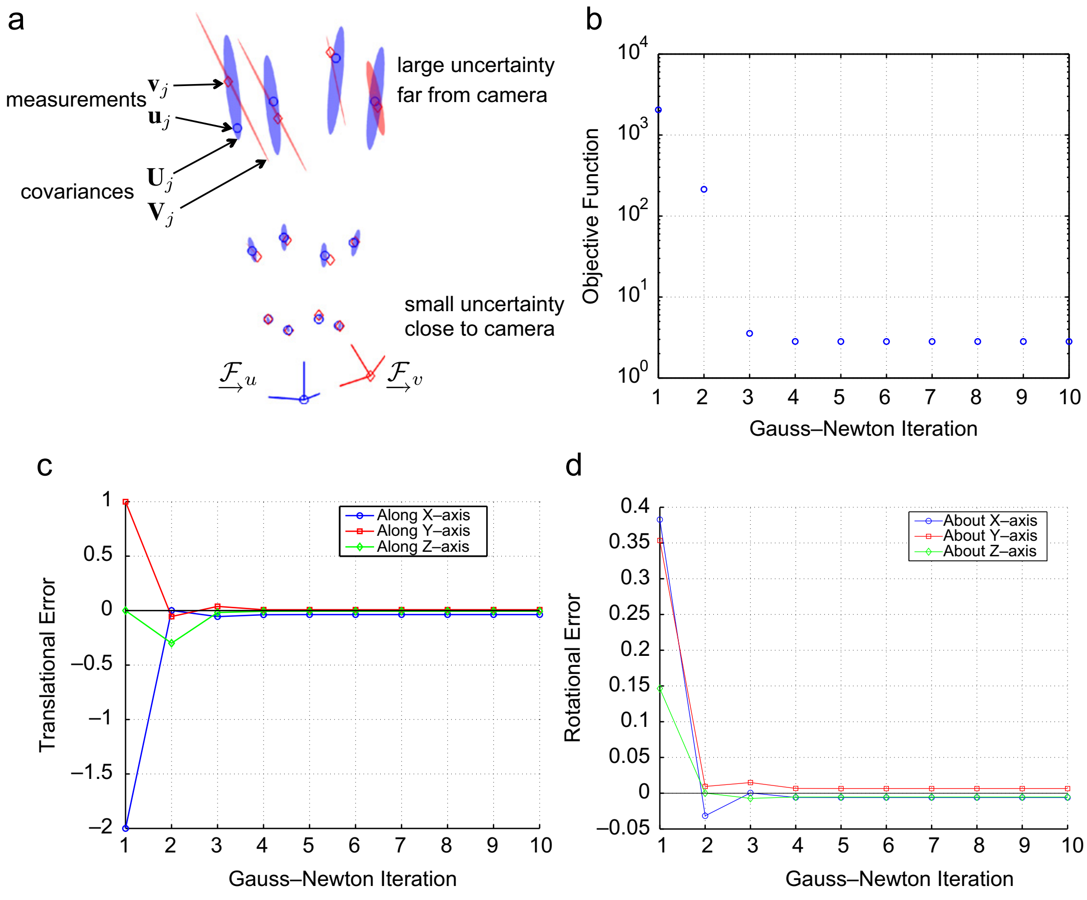

Fig. 3. Numerical example of point-cloud alignment. (a) Aligned stereo camera measurements of points, $\mathbf{p}_j$: $\mathbf{u}_j$ (blue circles), $\mathbf{v}_j$ (red diamonds), and 3σ-covariances. (b) Convergence of objective function, J. (c) Convergence of translational errors. (d) Convergence of rotational errors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

iterations of the Gauss–Newton procedure. Figs. 3(c) and (d) depict the translational and rotational errors (compared to the true values of **C** and **r**), respectively. We see that within a few iterations, the algorithm converges to its final value and that the resulting errors are acceptably small. Note, the errors do not converge to zero because the measurements are noisy and the problem is overconstrained (i.e., we have 12 measurements but require only three to solve for pose). This results in a residual error, as would any least-squares method.

5. Conclusion

This paper has presented a first-principles approach to linearizing expressions involving rotations represented by either 3×3 rotation matrices or 4×1 unit-length quaternions. The linearization approach was demonstrated through two examples: (i) linearizing a sun sensor measurement error term, and (ii) performing weighted-least-squares point-cloud alignment. Specifically, we believe the contributions of the paper are:

1. Compact matrix expressions for the partial derivative of both a rotation matrix and a unit-length quaternion with respect to constituent Euler angles, given by (22) and (40), respectively.
2. A first-principles derivation of the multiplicative constraint-sensitive perturbations of a rotation matrix and a unit-length quaternion, given by (26) and (44), respectively. These may be used to linearize any expression involving a rotation matrix or unit-length quaternion.
3. Expressions for updating rotation matrices and unit-length quaternions with a constraint-sensitive perturbation are provided in (31) and (49), respectively. These updates avoid the need to restore constraints afterwards.
4. Demonstration of linearizing a sun sensor measurement error term involving a unit-length quaternion. The resulting linearized error term, given by (61), may be used in a batch-style estimator in combination with other types of measurements for attitude determination.
5. Demonstration of the approach to linearizing expressions involving rotation matrices through an extended pose estimation example: weighted-least-squares point-cloud alignment. The procedure outlined is able to handle arbitrary motions (i.e., it has no singularities) and matrix weights.

We have shown that the multiplicative perturbations to rotation matrices and unit-length quaternions are a consequence of straightforward linearization with respect to a set of three Euler angles. This result can almost certainly be generalized to other three-parameter representations of rotations. Also, throughout the paper we have employed constraint-sensitive perturbations to rotations and then used corresponding update rules that automatically restore the constraints. Given the results of Zanetti et al. [32] for Kalman filtering, it may be possible to show that constraint restoration for batch-style estimation is also a consequence of the optimization procedure rather than assuming it a priori. Finally, an interesting avenue for future research might be to employ our approach to linearizing rotations in other types of problems than pose estimation, e.g., dynamics and control.

Acknowledgements

The authors would like to thank the Natural Sciences and Engineering Research Council (NSERC) of Canada for supporting this work through grants and scholarships.

References

[1] I.Y. Bar-Itzhack, Y. Oshman, Attitude determination from vector observations: quaternion estimation, IEEE Transactions on Aerospace and Electronic Systems AES-21 (1985) 128–136.
[2] D.C. Brown, A solution to the general problem of multiple station analytical stereotriangulation, Rca-mtp Data Reduction Technical Report no. 43 (or afmtc tr 58-8), Patrick Airforce Base, Florida, 1958.
[3] J.C.K. Chou, Quaternion kinematic and dynamic differential equations, IEEE Transactions on Robotics and Automation 8 (1) (1992) 53–64.
[4] J.L. Crassidis, F.L. Markley, Y. Cheng, A survey of nonlinear attitude estimation methods, Journal of Guidance, Control, and Dynamics 30 (1) (2007) 12–28.
[5] T.A. Ell, On Systems of Linear Quaternion Functions, ArXiv Mathematics e-prints, February 2007.
[6] W.R. Hamilton, Elements of Quaternions, Longmans, Green & Co., London, England, 1866.
[7] A.S. Hardy, Elements of Quaternions, Ginn & Company, Boston, USA, 1891.
[8] B. Horn, Closed-form solution of absolute orientation using unit quaternions, Journal of the Optical Society of America A 4 (4) (1987) 629–642.
[9] P.C. Hughes, Spacecraft Attitude Dynamics, John Wiley & Sons, New York, 1986.
[10] B.P. Ickes, A new method for performing digital control system attitude computations using quaternions, AIAA Journal 8 (1) (1970) 13–17.
[11] R.E. Kalman, A new approach to linear filtering and prediction problems, Transactions of the ASME—Journal of Basic Engineering 82 (Series D) (1960) 35–45.
[12] E.J. Lefferts, F.L. Markley, M.D. Shuster, Kalman filtering for spacecraft attitude estimation, Journal of Guidance, Control, and Dynamics 5 (5) (1982) 417–429.
[13] P. Mahalanobis, On the generalized distance in statistics, in: Proceedings of the National Institute of Science, vol. 2, India, 1936, pp. 49–55.
[14] M. Maimone, Y. Cheng, L. Matthies, Two years of visual odometry on the Mars exploration rovers, Journal of Field Robotics 24 (3) (2007) 169–186.
[15] F.L. Markley, Attitude error representations for Kalman filtering, Journal of Guidance, Control, and Dynamics 26 (2) (2003) 311–317.
[16] F.L. Markley, Attitude estimation or quaternion estimation?, Journal of the Astronautical Sciences 52 (1–2) (2004) 221–238
[17] F.L. Markley, Multiplicative vs. additive filtering for spacecraft attitude determination, in: Proceedings of the Sixth Conference on Dynamics and Control of Systems and Structures in Space (DCSSS), vol. D22, Riomaggiore, Italy, July 2004.
[18] L. Matthies, S.A. Shafer, Error modeling in stereo navigation IEEE Journal of Robotics and Automation 3 (3) (1987) 239–248.
[19] L.H. Matthies, Dynamic stereo vision, Ph.D. Thesis, Carnegie-Mellon University, 1989.
[20] J. Nocedal, S.J. Wright, Numerical Optimization, second ed., Springer, 2006.
[21] B. Peirce, Linear associative algebra, American Journal of Mathematics 4 (1) (1881) 97–229.
[22] M.D. Shuster, A survey of attitude representations, Journal of the Astronautical Sciences 41 (4) (1993) 439–517.
[23] M.D. Shuster, Constraint in attitude estimation part i: constrained estimation, Journal of the Astronautical Sciences 51 (1) (2003) 51–74.

[24] M.D. Shuster, Constraint in attitude estimation part ii: unconstrained estimation, Journal of the Astronautical Sciences 51 (1) (2003) 75–101.
[25] R.C. Smith, P. Cheeseman, On the representation and estimation of spatial uncertainty, The International Journal of Robotics Research 5 (4) (1986) 56–68.
[26] R.C. Smith, M. Self, P. Cheeseman, Estimating uncertain spatial relationships in robotics, in: I.J. Cox, G.T. Wilfong (Eds.), Autonomous Robot Vehicles, Springer Verlag, New York, 1990, pp. 167–193.
[27] J. Stuelpnagel, On the parameterization of the three-dimensional rotation group, SIAM Review 6 (4) (1964) 422–430.
[28] P.G. Tait, An Elementary Treatise on Quaternions, second ed., Cambridge University Press, 1873.
[29] B. Triggs, P.F. McLauchlan, R.I. Hartley, A.W. Fitzgibbon, Bundle adjustment—a modern synthesis, in: B. Triggs, A. Zisserman, R. Szeliski (Eds.), Vision Algorithms '99, Lecture Notes in Computer Science, vol. 1883, Springer-Verlag, Berlin, Heidelberg, 2000, pp. 298–372.
[30] G. Wahba, Problem 65-1: a least squares estimate of satellite attitude, SIAM Review 7 (3) (1965) 409.
[31] E.T. Whittaker, A Treatise on the Analytical Dynamics of Particles and Rigid Bodies, fourth ed., Cambridge University Press, 1959.
[32] R. Zanetti, M. Majji, R. Bishop, D. Mortari, Norm-constrained Kalman filtering, Journal of Guidance, Control, and Dynamics 32 (5) (2009) 1458–1465.