ADDENDUM 2: OPERATIONS WITH QUATERNIONS

PANTELIS SOPASAKIS AND PANOS PATRINOS

Abstract. This document is to repeat certain important facts regarding operations with quaternions that might have eluded your attention. In particular, certain operations such as *addition* and *subtraction* should be avoided. Instead, operations such as Hamilton products and cojugates possess favourable properties.

Contents

1. Operations with quaternions

1.1. **General remarks.** Let $\mathfrak{q} \in \mathbb{H}_1$ be a **unit quaternion** which is written as

$$\mathfrak{q} = \begin{bmatrix} q_0 \\ q \end{bmatrix}, \tag{1.1}$$

where $q_0 \in \mathbb{R}$ is its **scalar part** and $q \in \mathbb{R}^3$ is its **vector part**. The norm of $\mathfrak{q}$ is $\|\mathfrak{q}\| = \sqrt{q_0^2 + \|q\|^2}$. Recall that a quaternion is called unitary if $\|\mathfrak{q}\| = 1$. In that case $q_0 = \pm\sqrt{1 - \|q_0\|^2}$. The quaternion $\mathfrak{q} = (1, 0, 0, 0)$ is called the **no-rotation** quaternion. In certain cases, we may assume that $q_0 = +\sqrt{1 - \|q_0\|^2}$, however, we need to be very careful with this assumption. For example, we used it to derived a reduced state-space model in `ANC-quaternion-model.pdf` (see version 1.3.9 or newer). Whenever we use this assumption we must verify its validity.

Certain operations with quaternions should be avoided. First addition of quaternions (treating them as vectors):

(1) Given two quaternions $\mathfrak{q}, \mathfrak{p} \in \mathbb{H}_1$ with $\mathfrak{q} = (q_0, q)$ and $\mathfrak{p} = (p_0, p)$, the sum $\mathfrak{q} + \mathfrak{p} = (q_0 + p_0, q + p)$ is certainly not a unit quaternion,
(2) If $\mathfrak{q} \in \mathbb{H}_1$ is any unit quaternion and $\mathfrak{p} = (1, 0, 0, 0)$ is the no-rotation quaternion, then $\mathfrak{q} + \mathfrak{p}$ is still not a unit quaternion,
(3) If $\mathfrak{q}$ and $\mathfrak{p}$ are two quaternions and p and q are their vector parts, then take the 3-dimension vector $p + q$. Is this the vector part of a quaternion? It is not clear how we could construct a unit quaternion out of $p + q$. For example $\sqrt{1 - \|p + q\|^2}$ might well be an imaginary number.

The same holds for differences of quaternions as explained in the course slides.

(1) Given two quaternions $\mathfrak{p}, \mathfrak{q} \in \mathbb{H}_1$, $\mathfrak{p} - \mathfrak{q} = (p_0 - q_0, p - q)$ is not a unit quaternion. For example, the quaternions $\mathfrak{p} = (1, 0, 0, 0)$ and $\mathfrak{q} = (1, 0, 0, 0)$ are equal to one another, but $\mathfrak{p} - \mathfrak{q} = (0, 0, 0, 0)$. Not only is this not unitary, but it does not represent the no-rotation quaternion as we would expect from a well-behaved difference operator.

(P. Sopasakis and P. Patrinos) KU Leuven, Department of Electrical Engineering (ESAT), STADIUS Center for Dynamical Systems, Signal Processing and Data Analytics & Optimization in Engineering (OPTEC), Kasteelpark Arenberg 10, 3001 Leuven, Belgium. Email addresses: pantelis.sopasakis@kuleuven.be and panos.patrinos@esat.kuleuven.be.

Lecture Notes for the 3rd-year BSc project “EAGLE” at KU Leuven. version 1.0.0. Last updated: November 21, 2017.

(2) Say we have two quaternions $\mathfrak{p}, \mathfrak{q} \in \mathbb{H}_1$ with $\mathfrak{p} - \mathfrak{q} = (-1.8874, -0.5371, 0.2898, -0.2554)$. At a first glance we should infer that the two quaternions are different from one another because their difference is not equal to the no-rotation quaternion (or to zero). Take, however, $\mathfrak{q} = (0.9437, 0.2685, -0.1449, 0.1277)$ and $\mathfrak{p} = -\mathfrak{q}$ (these both correspond to the same orientation with a yaw of $10°$, a pitch of $-20°$ and a roll of $30°$).

Now, recall the **Hamilton product** of two quaternions $\mathfrak{p}, \mathfrak{q} \in \mathbb{H}_1$, that is

$$\mathfrak{p} \otimes \mathfrak{q} = \begin{bmatrix} p_0 q_0 - p \cdot q \\ p_0 q + q_0 p + p \times q \end{bmatrix} \in \mathbb{H}_1. \tag{1.2}$$

Recall also that for $\mathfrak{p} \in \mathbb{H}_1$, the **conjugate** quaternion is defined as

$$\mathfrak{p}^* = (p_0, -p). \tag{1.3}$$

Using the conjugate quaternion, we may define the **difference** of two quaternions as

$$\delta(\mathfrak{p}, \mathfrak{q}) = \mathfrak{p} \otimes \mathfrak{q}^*. \tag{1.4}$$

What is very important to observe here is that for any $\mathfrak{p}, \mathfrak{q} \in \mathbb{H}_1$, $\mathfrak{p}^* \in \mathbb{H}_1$, $\mathfrak{p} \otimes \mathfrak{q} \in \mathbb{H}_1$ and $\delta(\mathfrak{p}, \mathfrak{q}) \in \mathbb{H}_1$, meaning, the results of these operations are always unit quaternions.

2. Important Remarks

2.1. **Using the full-quaternion model.** Before we proceed, we believe it is very important to repeat that when **simulating** the system, we must use the **full-quaternion model**. We used a reduced model (which makes use of the vector part of the quaternion) in order to linearise the system and design an LQR (a linear gain K) and a Kalman filter (the observer gain L). Unless we use the full-quaternion model (with the 4-dimensional quaternion) we might be disregarding flips ($180°$ rotations) which are of course undesirable.

2.2. **Imposing noise on quaternions.** The first implication of this observation is when you simulate your system and you want to impose state noise on the quaternion part of the state, **do not simply add noise** because the result will not be a quaternion any more. In other words, do not do:

$$\begin{bmatrix} \mathfrak{q}_{k+1} \\ \omega_{k+1} \\ n_{k+1} \end{bmatrix} \leftarrow F\left(\begin{bmatrix} \mathfrak{q}_k \\ \omega_k \\ n_k \end{bmatrix}, u_k \right) + w_k, \tag{2.1}$$

where w_k is some noise term and F is the discrete-time nonlinear dynamics (implemented using `ode45` in MATLAB). It is equally wrong to do

$$\begin{bmatrix} q_{k+1} \\ \omega_{k+1} \\ n_{k+1} \end{bmatrix} \leftarrow F_{\text{reduced}}\left(\begin{bmatrix} q_k \\ \omega_k \\ n_k \end{bmatrix}, u_k \right) + w_k, \tag{2.2}$$

where q is the vector part of $\mathfrak{q}$ and F_{reduced} is the reduced-order nonlinear system dynamics. Both these things are wrong. We **cannot** add quaternions. The **correct** way to go is

$$\begin{bmatrix} \mathfrak{q}_{k+1} \\ \omega_{k+1} \\ n_{k+1} \end{bmatrix} \leftarrow F\left(\begin{bmatrix} \mathfrak{q}_k \\ \omega_k \\ n_k \end{bmatrix}, u_k \right), \tag{2.3a}$$

$$\mathfrak{q}_{k+1} \leftarrow \mathfrak{q}_{k+1} \otimes \delta\mathfrak{q}_k, \tag{2.3b}$$

$$\omega_{k+1} \leftarrow \omega_{k+1} + \delta\omega_k, \tag{2.3c}$$

$$n_{k+1} \leftarrow n_{k+1} + \delta n_k, \tag{2.3d}$$

where $\delta\mathfrak{q}_k$ is a (small) quaternion noise term and $\delta\omega_k$ is a noise term acting on ω and δn_k is a noise term acting on n.

2.3. **Reference tracking the right way.** This is about reference tracking. Using the reference tracking v2 methodology described in the course material (see slide 50/177, course slides v3.1.24), the control action is computed by

$$u_k = u^e + K(x - x^e). \tag{2.4}$$

Here, we need to replace the minus by an appropriate quaternion difference. Before we proceed, let us write down (2.4) in an equivalent form

$$e_k = x - x^e, \tag{2.5a}$$

$$u_k = u^e + K e_k, \tag{2.5b}$$

where e_k denotes the **state error**. As you may have already guessed, when our dynamical system involves quaternions, it is not a good idea to use a minus there. Instead, we should replace it by an appropriate **difference operator** as explained above.

The correct way to go is

$$e_k^q = \delta(\mathfrak{q}_k, \mathfrak{q}_k^e), \tag{2.6a}$$

$$e_k^\omega = \omega_k - \omega_k^e, \tag{2.6b}$$

$$e_k^n = n_k - n_k^e, \tag{2.6c}$$

where $\mathfrak{q}_k^e$ is the reference quaternion (i.e., the reference orientation) and $\omega_k^e = 0$ (the reference angular velocity) and $n_k^e = 0$ (the reference deviation from the hovering spin of rotation).

There is a little caveat here and that is that in practice we do not use $\mathfrak{q}_k$, but we instead have an estimate of the vector part quaternion $\hat{q}_k$. This means that we lack knowledge of $\hat{q}_{0,k}$ (the scalar part of the quaternion). We only know that

$$\hat{q}_{0,k} = \pm\sqrt{1 - \|\hat{q}_k\|^2}, \tag{2.7}$$

but we don't know whether we should use the positive or the negative square root. Nevertheless, we may use the sign of the scalar part of the **measured quaternion** $\mathfrak{q}_k$. By doing so, we will have

$$\hat{q}_{0,k} = \operatorname{sign}(q_{0,k})\sqrt{1 - \|q_k\|^2}. \tag{2.8}$$

2.4. **Normalization is necessary.** In our simulations, every time we use `ode45` — or any other solver — to simulate the continuous-time nonlinear system from time kT_s to time $(k+1)T_s$ we expect that $\mathfrak{q}_{k+1}$ is unitary. Indeed, the solutions of the system $\dot{\mathfrak{q}} = \frac{1}{2}\mathfrak{q} \otimes (0, \omega)$ have the property $\mathfrak{q}(t) \in \mathbb{H}_1$ for all $t \in \mathbb{R}$. Nevertheless, `ode45` does not solve the system totally accurately, therefore, we should not expect that $\|\mathfrak{q}_{k+1}\| = 1$[1]. As a result, errors might build up leading to erroneous results. In order to avoid that, when we run simulations we need to normalize the quaternion after every such invocation to `ode45`

$$\mathfrak{q}_{k+1} \leftarrow \frac{\mathfrak{q}_{k+1}}{\|\mathfrak{q}_{k+1}\|}. \tag{2.9}$$

[1]It will be $\|\mathfrak{q}_{k+1}\| \approxeq 1$, but not $\|\mathfrak{q}_{k+1}\| = 1$.