AN INTRODUCTION TO USING DUAL QUATERNIONS TO STUDY KINEMATICS

STEPHEN MONTGOMERY-SMITH AND CECIL SHY

Abstract. We explain the use of dual quaternions to represent poses, twists, and wrenches.

1. Introduction

We assume that the reader is familiar with the notion of *pose* and *rigid motion*, and the use of quaternions to represent rotations. We use the terms pose and rigid motion interchangeably, since a pose can be considered as a rigid motion relative to a fixed reference frame, and the mathematical operations to manipulate them are identical. Thus we also use the terms rotation and orientation interchangeably, and similarly with translation and position. We refer the reader to [5, 9, 21, 25].

The use of dual quaternions to represent poses is well established,: [1, 2, 7, 8, 10, 12, 13, 14, 15, 17, 19, 20, 24, 27, 30].

Let us start with a general discussion of why people might want to represent rotations by unit quaternions instead of three by three orthogonal matrices with determinant 1. One reason given is that quaternions can be represented with only four numbers, whereas the matrix representation requires nine numbers. Another representation commonly used are Euler angles, and these only require three numbers. However, with Euler angles, the formula for composition of rotations is difficult to calculate. Also, Euler angles suffer from gimbal lock, in which if the rotation takes certain values, the choice of Euler angles isn't unique, and discontinuities and singularities arise. Quaternions suffer from no such problems.

But in modern times, computer memory is very cost effective, and communication is fast, thus the aforementioned advantages might seem trivial at first. This begs the question: what other advantages do quaternions offer over matrices? Quaternions represent a quantity with three degrees of freedom by a four dimensional vector, whereas matrices use a nine dimensional vector. Suppose one obtains an approximation

Date: July 22, 2023.

to a matrix or quaternion? For example, this would be common when using Newton-Raphson to find a rotation satisfying some formula, or to interpolate between rotations. So then one has to find a projection which takes the vector to an admissible vector, that is, a vector that actually represents a rotation. And projecting from a nine dimensional quantity to a three dimensional quantity is going to be much harder than the similarly projection from the four dimensional vector. (The projection from quaternions to unit quaternions is called the *normalization*, and is defined in Section2. A suggestion of what such a projection would be for (3×3) matrices is given at the end of Section 10.)

For representing poses, the usual representation is by a three by three orthogonal matrix with determinant 1 for the rotation, and a three dimensional vector for the translation, twelve dimensional in all. This has a natural representation as a four by four matrix, as we explain Section 4. A great advantage of this representation is the relationship between pose and twist:

$$\frac{d}{dt}(\text{pose}) = (\text{pose}) \times (\text{twist}). \tag{1}$$

where the multiplication is matrix multiplication, and a twist is a combination of angular velocity and translational velocity measured with respect to the moving frame.

Another natural representation used a lot is to use a quaternion and a three dimensional vector. But this new representation has problems in that it is not clear how the above equation would work in this situation. However, if one introduces the notion of the dual quaternion, which is an eight dimensional vector, equation (1) is still valid if one uses dual quaternion multiplication.

A challenge with dual quaternions is that they are abstract, which can make them harder to understand. Mathematicians find value behind abstract computational ideas, but Engineers usually find the end result, 'black box approach,' more useful.

A useful approach to developing intuition for dual quaternions is to understand the rules with their structure, and the mathematical operations needed to manipulate them, (that is, normalization, basis vector multiplication, pose differentiation). Once one becomes comfortable with their form and function, the advantage of duel quaternions over other methods becomes more apparent. For example, one realizes that computing the projection of an eight dimensional vector to the six degrees of freedom of a pose, is also quite simple. (This projection is also called the normalization, and is defined in Section 3.)

This makes it very useful if one is using the Newton-Raphson Method to find a pose satisfying certain properties. We show how the dual quaternion representation efficiently solves the forward kinematics problem.

Finally we consider the dynamics of poses, that is, the equations of motion of a rigid spinning body. In any representation this can be quite difficult, because of both centripetal and precessional effects. But from a theoretical perspective, using the dual quaternion representation is quite slick.

In summary, while the usual representation using Euler angles and translations is conceptually easy to understand, performing calculations can be difficult. Whereas with dual quaternions, while the are initially conceptually hard to understand, when it comes to calculations they are much easier to work with.

2. Motivation

A *pose*, or *rigid motion*, is a rotation followed by a translation. (Semantically, a pose is the position of the end effector of the robot, and thus different from a rigid motion, but mathematically they are the same if we consider a pose to be a rigid motion with respect to the stationary reference frame. But see [6] for a different point of view.) The rotation is represented by a (3×3) orthogonal matrix R with determinant 1. A translation $\boldsymbol{t}$ is a vector in $\mathbb{R}^3$. These represent the pose $(\mathsf{R}, \boldsymbol{t})$:

$$\boldsymbol{r} \mapsto \mathsf{R}\boldsymbol{r} + \boldsymbol{t}. \tag{2}$$

Composition of poses is written from right to left. Thus

$$(\mathsf{R}_1, \boldsymbol{t}_1) \circ (\mathsf{R}_2, \boldsymbol{t}_2) = (\mathsf{R}_1\mathsf{R}_2, \boldsymbol{t}_1 + \mathsf{R}_1\boldsymbol{t}_2). \tag{3}$$

One way to represent a pose is using a four by four matrix

$$\begin{bmatrix} \mathsf{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathsf{R}_{11} & \mathsf{R}_{12} & \mathsf{R}_{13} & t_1 \\ \mathsf{R}_{21} & \mathsf{R}_{22} & \mathsf{R}_{23} & t_2 \\ \mathsf{R}_{31} & \mathsf{R}_{32} & \mathsf{R}_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

Then composition of poses may be computed simply by multiplying matrices of the form (4). In the literature, the Lie group of rotations is often denoted $SO(3)$. The Lie group of poses is often denoted $SE(3)$.

The problem with this representation is that twelve numbers are required to represent a pose. Other problems, which we discuss below, is that of interpolating a sequence of poses, and of computing forward kinematics.

One common way to solve both of these problems is to represent the rotation by a unit quaternion [3, 18], which we briefly describe here. A *quaternion* is a quadruple of real numbers, written as $A = w + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}$, with the algebraic operations $\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1$. Its *conjugate* is $A^* = w - x\boldsymbol{i} - y\boldsymbol{j} - y\boldsymbol{k}$, its *norm* is $|A| = (w^2 + x^2 + y^2 + z^2)^{1/2} = \sqrt{AA^*} = \sqrt{A^*A}$, its *normalization* is $\widehat{A} = A/|A|$, its *real part* is $\mathrm{Re}(A) = w = \frac{1}{2}(A + A^*)$, and its *imaginary part* is $\mathrm{Im}(A) = \boldsymbol{i}x+\boldsymbol{j}y+\boldsymbol{k}z = \frac{1}{2}(A-A^*)$. It is called a *unit* quaternion if $|A| = 1$, a *real* quaternion if $\mathrm{Im}(A) = 0$, and a *vector* quaternion if $\mathrm{Re}(A) = 0$. Note the multiplicative inverse is given by $A^{-1} = A^*/|A|^2$.

We identify three dimensional vectors with vector quaternions, by identifying $\boldsymbol{i}$, $\boldsymbol{j}$, and $\boldsymbol{k}$ with the three standard unit vectors. A unit quaternion Q represents the rotation $\boldsymbol{r} \mapsto Q\boldsymbol{r}Q^*$. A rotation by angle a about an axis $\boldsymbol{s}$, where $|\boldsymbol{s}| = 1$, has two unit quaternion representations: $\pm(\cos(\frac{1}{2}a) + \boldsymbol{s}\sin(\frac{1}{2}a))$. Composition of rotations corresponds to multiplication of unit quaternions. With some practice, it becomes easier to read a rotation from a quaternion than it does from Euler angles, the standard method of representing rotations.

We can represent quaternions as four dimensional vectors, and give it the inner product

$$A \cdot B = \mathrm{Re}(AB^*) = \mathrm{Re}(A^*B). \tag{5}$$

Then a way to represent poses is as a pair $(Q, \boldsymbol{t})$. This is the way poses are represented internally in the Robotic Operating System [26]. Then the composition rule is

$$(Q_1, \boldsymbol{t}_1) \circ (Q_2, \boldsymbol{t}_2) = (Q_1Q_2, \boldsymbol{t}_1 + Q_1\boldsymbol{t}_2Q_1^*). \tag{6}$$

A difficulty with this is that the composition is not a bilinear operation as it was with the representation (4). (This lack of bilinear representation becomes particularly problematic when representing twists, which we describe below.) However there is a way to resolve this as follows. Write $B_1 = \frac{1}{2}\boldsymbol{t}_1Q_1$ and $B_2 = \frac{1}{2}\boldsymbol{t}_2Q_2$. Then equation (6) can be rewritten as

$$\begin{aligned} &(Q_1, 2B_1Q_1^{-1}) \circ (Q_2, 2B_2Q_2^{-1}) \\ &= (Q_1Q_2, 2B_1Q_1^{-1} + 2Q_1B_2Q_2^{-1}Q_1^{-1}) \\ &= (Q_1Q_2, 2(B_1Q_2 + Q_1B_2)(Q_1Q_2)^{-1}). \end{aligned} \tag{7}$$

This leads to representing the pose $(Q, \boldsymbol{t})$ by a unit dual quaternion, which we now describe.

3. Dual quaternions to represent poses

A *dual quaternion* is a pair of quaternions, written as $\eta = A + \epsilon B$, with the extra algebraic operation $\epsilon^2 = 0$.

The *conjugate* dual quaternion of $\eta = \eta = A + \epsilon B$ is $\eta^* = A^* + \epsilon B^*$. Conjugation reverses the order of multiplication:

$$(\eta_1 \eta_2)^* = \eta_2^* \eta_1^*. \tag{8}$$

There is another conjugation for dual quaternions: $\overline{(A + \epsilon B)} = A - \epsilon B$, but we have no cause to use it in this paper, except in equation (18) below.

A *unit* dual quaternion $\eta = Q + \epsilon B$ is a dual quaternion such that $\eta\eta^* = 1$, equivalently, that Q is a unit quaternion and $B \cdot Q = 0$. A *vector* dual quaternion $A + \epsilon B$ is a dual quaternion such that both A and B are vector quaternions.

If $\eta = Q + \epsilon B$ is a dual quaternion with $Q \neq 0$, then its multiplicative inverse can be calculated using the formula

$$\eta^{-1} = Q^{-1} - \epsilon Q^{-1} B Q^{-1}. \tag{9}$$

If η is a unit dual quaternion, then there is a computationally much faster formula:

$$\eta^{-1} = \eta^*. \tag{10}$$

A *dual number* is anything of the form $a + \epsilon b$, where a and b are real numbers. The *norm* of a dual quaternion $\eta = Q + \epsilon B$ is the dual number defined by the two steps:

$$|\eta|^2 = \eta^* \eta = \eta\eta^* = |Q|^2 + 2\epsilon(B \cdot Q), \tag{11}$$

$$|\eta| = \sqrt{|\eta|^2} = |Q| + \epsilon(B \cdot Q)/|Q|. \tag{12}$$

The norm preserves multiplication, that is, if η_1 and η_2 are two dual quaternions, then

$$|\eta_1 \eta_2| = |\eta_1||\eta_2|. \tag{13}$$

If $\eta = Q + \epsilon B$ is any dual quaternion with $Q \neq 0$, then we define its *normalization* to be the unit dual quaternion

$$\widehat{\eta} = |\eta|^{-1}\eta = \eta|\eta|^{-1} = Q/|Q| + \epsilon(B - (B \cdot Q)Q/|Q|^2). \tag{14}$$

(We remark that the normalization of a dual quaternion is used in the computer graphics industry [12, 13].) While this normalization formula might seem initially quite complicated, after thinking about it one can see that it is the simplest projection that enforces $|Q| = 1$ and $B \cdot Q = 0$.

The normalization also satisfies the following properties.

- If η is a unit dual quaternion, then $\widehat{\eta} = \eta$.

- Normalization preserves multiplication, that is, if η_1 and η_2 are two dual quaternions, then

$$\widehat{\eta_1\eta_2} = \widehat{\eta}_1\widehat{\eta}_2. \tag{15}$$

Note that

$$\eta^{-1} = |\eta|^{-2}\eta^*, \tag{16}$$

and this could have been the definition of the multiplicative inverse of a dual quaternion, except that this definition is circular.

We set $\mathfrak{dh}$ for the set of dual quaternions, $\mathbb{SDH}$ for the set of unit dual quaternions, and $\mathfrak{sdh}$ for the set of vector dual quaternions.

If we are given a pose represented by $(Q, \boldsymbol{t})$, then the pose is also represented by the unit dual quaternion

$$\eta = Q + \tfrac{1}{2}\epsilon \boldsymbol{t} Q. \tag{17}$$

As we have shown above, composition of poses corresponds to multiplication of unit dual quaternions. If $\boldsymbol{r}$ is a 3-vector, and $\boldsymbol{s}$ is the image of $\boldsymbol{r}$ under the action of the pose $\eta = Q + \epsilon B$, then

$$1 + \epsilon \boldsymbol{s} = \eta(1 + \epsilon \boldsymbol{r})\overline{\eta}^*, \tag{18}$$

but generally it is easier to use the formula

$$\boldsymbol{s} = Q\boldsymbol{r}Q^* + 2BQ^* = (Q\boldsymbol{r} + 2B)Q^*. \tag{19}$$

The introduction of the factors 2 and $\frac{1}{2}$ is unnecessary, but it slightly improves formulas for the twist, as we see below.

It is difficult for a human to look at the second part of a unit dual quaternion, and from that read what the translational part of the pose should be. So dual quaternions work better for an internal representations of poses rather than human readable representations.

4. Kinematics: dual quaternions to represent twists

The translational velocity $\boldsymbol{v}$ of a translating reference frame is given by $\boldsymbol{v} = \dot{\boldsymbol{t}}$. The angular velocity $\boldsymbol{w} = (w_1, w_2, w_3)$ of a rotating frame is given by the differential equation

$$\dot{\mathsf{R}} = \mathsf{R}\mathsf{S}(\boldsymbol{w}), \tag{20}$$

or $\mathsf{S}(\boldsymbol{w}) = \mathsf{R}^{-1}\dot{\mathsf{R}}$, where $\mathsf{S}(\boldsymbol{w})$ is the *Hodge star operator* of $\boldsymbol{w}$ [29]:

$$\mathsf{S}(\boldsymbol{w}) = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix}. \tag{21}$$

(Equation (20) gives the angular velocity with respect to the rotating reference frame. If one wants the angular velocity with respect to the stationary reference frame, use instead $\dot{\mathsf{R}} = \mathsf{S}(\boldsymbol{w})\mathsf{R}$.)

Note that

$$\mathsf{S}(\boldsymbol{w})\boldsymbol{r} = \boldsymbol{w} \times \boldsymbol{r}, \tag{22}$$

where $\times$ denotes the cross product.

A *twist* is the pair of vectors $(\boldsymbol{w}, \boldsymbol{v})$ that describe the change of pose in the moving reference frame, that is:

$$\frac{d}{dt}\begin{bmatrix}\mathsf{R} & \boldsymbol{t}\\ 0 & 1\end{bmatrix} = \begin{bmatrix}\mathsf{R} & \boldsymbol{t}\\ 0 & 1\end{bmatrix}\begin{bmatrix}\mathsf{S}(\boldsymbol{w}) & \boldsymbol{v}\\ 0 & 0\end{bmatrix}. \tag{23}$$

(Note in particular that $\dot{\boldsymbol{t}} = \mathsf{R}\boldsymbol{v}$, so that a continual application of a twist can result in a motion following a straight line, an arc of a circle, or a spiral.) The set of angular velocities, and the set of twists are both Lie algebras, often denoted by $\mathfrak{so}(3)$ and $\mathfrak{se}(3)$ in the literature.

The quaternion-translation formulation doesn't have a representation like equation (1):

$$\frac{d}{dt}(Q, \boldsymbol{t}) = \left(\tfrac{1}{2}Q\boldsymbol{w}, Q\boldsymbol{v}Q^*\right). \tag{24}$$

But with the unit dual representation, because the composition of poses is given by a bilinear multiplication, it follows that the twist is represented by the dual quaternion φ satisfying the differential equation.

$$\varphi = \eta^{-1}\dot{\eta}, \quad \text{or} \quad \dot{\eta} = \eta\varphi. \tag{25}$$

It will be shown in Section 9 (see also [10, 27]) that φ is the vector dual quaternion

$$\varphi = \tfrac{1}{2}\boldsymbol{w} + \tfrac{1}{2}\epsilon\boldsymbol{v}. \tag{26}$$

Note that if η is a unit dual quaternion that is a function of time t, then $\eta^{-1}\dot{\eta}$ is always a vector dual quaternion. (Note that $\dot{\eta}\eta^{-1}$ is a vector dual quaternion equal to the twist relative to the fixed frame of reference. Note also that the factor $\frac{1}{2}$ attached to $\boldsymbol{v}$ in equation (26) is a consequence of the introduction of the factor $\frac{1}{2}$ in equation (17), and thus is essentially arbitrary.)

5. Dual quaternions to represent wrenches

Let the pose η represent the reference frame that moves with the end effector. It is not necessary (although it can simplify things) that the center of mass of the end effector coincides with the origin of the moving frame.

The *wrench dual quaternion* is defined to be

$$\tau = 2\boldsymbol{q} + 2\epsilon\boldsymbol{p}, \tag{27}$$

where $\boldsymbol{q}$ and $\boldsymbol{p}$ are the torque and force, respectively, applied to the end effector at the origin of the moving frame, measured with respect to the moving frame.

If $\boldsymbol{r}_0$ is the center of mass of the end effector in the moving frame, then the twist about the center of mass is given by

$$\varphi_0 = \varphi + \tfrac{1}{2}\epsilon \boldsymbol{w} \times \boldsymbol{r}_0, \tag{28}$$

where $\varphi = \eta^{-1}\dot{\eta}$, and the wrench applied about the center of mass is

$$\tau_0 = \tau + 2\boldsymbol{p} \times \boldsymbol{r}_0. \tag{29}$$

The reason for introducing the factor 2 in definition (27) is so that the rate of change of work done to the end effector is given by

$$\frac{d}{dt}(\text{work done}) = \tau \cdot \varphi = \tau_0 \cdot \varphi_0. \tag{30}$$

See [4] for the origins of the term twist and wrench as pairs of 3-vectors, which are examples of *screws*.

6. Interpolation of poses

Suppose that we are given a sequence of times and rotations, $t_0 < t_1 < \cdots < t_n$ and $\mathsf{R}_0, \mathsf{R}_1, \ldots, \mathsf{R}_n$. We would like to find a function F from $[t_1, t_n]$ to rotations such that the third derivative of F is bounded, and $\mathsf{F}(t_k) = \mathsf{R}_k$. If the quantities $\mathsf{R}_0, \mathsf{R}_1, \ldots, \mathsf{R}_n$ were merely vectors, we could use cubic splines. But if we perform a similar calculation on the matrices, we cannot guarantee that the matrix $\mathsf{F}(t)$ is a rotation matrix.

So what we want is to find a map that projects general matrices onto rotation matrices. But it is not obvious how to find such a map that is computationally efficient. It is here that using quaternions really has great advantages over matrices, because an obvious map from general quaternions to unit quaternions is to normalize.

Let $Q_0, Q_1, \ldots, Q_n$ be the quaternion representation of $R_0, R_1, \ldots, R_n$ respectively. Let $G(t)$ be the cubic spline interpolation such that $G(t_k) = Q_k$. Then

$$F(t) = \frac{1}{|G(t)|} G(t) \tag{31}$$

provides a computationally fast, low jerk, method of interpolating the rotations.

This can be extended to interpolate poses. Given times $t_0 < t_1 < \cdots < t_n$ and unit dual quaternions $\eta_0, \eta_1, \ldots, \eta_n$, first interpolate to get a function γ such that $\gamma(t_k) = \alpha_k$, and then normalize it to get

a function $\eta(t) = \widehat{\gamma(t)}$. This gives a computationally fast, low jerk, method for interpolating poses.

It is perhaps better to interpolate the rotation quaternions and the translations separately. This is because the center of gravity typically travels in a straight line rather than the arc of a curve, but naive interpolation of dual quaternions follows arcs of curves if the angular rotation is non-zero. (Interpolation of dual quaternions is more appropriate for 'skinning' [12, 13].)

If one wants to find a function that passes through these points, and has low jerk even at the end points t_0 and t_n, this can be done by adding a 'ramp up' and 'ramp down' at the ends of the trajectory. Create a free cubic spline $\gamma(t)$ on $\frac{1}{2}(t_0 + t_1) < t_1 \cdots < t_{n-1} < \frac{1}{2}(t_{n-1} + t_n)$ and $\eta_0, \eta_1, \ldots, \eta_n$. Then define a function $s : \mathbb{R} \to [\frac{1}{2}(t_0 + t_1), \frac{1}{2}(t_{n-1} + t_n)]$

$$s(t) = \begin{cases} \frac{1}{2}(t_0 + t_1) & \text{if } t < t_0 \\ t_0 + (t_1 - t_0) f\left(\frac{t - t_0}{t_1 - t_0}\right) & \text{if } t_0 \leq t < t_1 \\ t & \text{if } t_1 \leq t < t_{n-1} \\ t_n - (t_n - t_{n-1}) f\left(\frac{t_n - t}{t_n - t_{n-1}}\right) & \text{if } t_{n-1} \leq t < t_n \\ \frac{1}{2}(t_{n-1} + t_n) & \text{if } t \geq t_n \end{cases} \tag{32}$$

where

$$f(t) = \tfrac{1}{2} + \tfrac{1}{2}(2 - t)t^3. \tag{33}$$

Since $f(0) = \frac{1}{2}$, $f'(0) = f''(0) = f''(1) = 0$, and $f(1) = f'(1) = 1$, it follows that $s(t)$ has bounded third derivative, $s(t_k) = t_k$ for $1 \leq k \leq n - 1$, $s(t) = \frac{1}{2}(t_0 + t_1)$ for $t \leq t_0$, and $s(t) = \frac{1}{2}(t_{n-1} + t_n)$ for $t \geq t_n$. Thus

$$\eta(t) = \widehat{\gamma(s(t))} \tag{34}$$

provides a low jerk function, even at the end points, that passes through the appropriate points. it is important that $t_1 - t_0$ and $t_n - t_{n-1}$ be large enough to allow the trajectory to 'ramp up' and 'ramp down' smoothly.

7. Perturbations of poses

Suppose one has a pose $\eta = \eta(t)$ that varies in time, and a fixed time t_0. Then we would like to find a good representation of the perturbation of $\eta(t)$ from a reference pose $\eta_r = \eta(t_0)$ when t is close to t_0. One way to do this is to choose a new reference frame in which $\eta_r = 1$, the identity pose, or rather, to think about how $\eta_r^{-1}\eta$ behaves.

It can be seen that for t close to t_0 that there is a vector dual quaternion θ such that

$$\eta = \eta_r \widehat{(1+\theta)} \approx \eta_r(1+\theta). \tag{35}$$

Thus

$$\theta \approx \eta_r^{-1}\eta - 1 = \eta_r^*\eta - 1 \tag{36}$$

gives a great way to measure the perturbation.

We prefer instead to use what we shall call the *Lie difference*, which we denote using $\triangle$, which is close to (36), but is always a vector dual quaternion:

$$\theta \approx \eta \triangle \eta_r = \tfrac{1}{2}(\eta_r^*\eta - \eta^*\eta_r) = \operatorname{Im}(\eta_r^*\eta). \tag{37}$$

Note that if θ_1 and θ_2 are vector dual quaternions, and η is an invertible dual quaternion, then

$$(\eta(1+\theta_1)) \triangle (\eta(1+\theta_2)) = \theta_1 - \theta_2 + O(|\theta_1|^2 + |\theta_2|^2). \tag{38}$$

This gives a great way to map perturbations of non-linear poses to linear twists. These can be very effective in control theory.

8. Spherical linear interpolation of dual quaternions

Suppose we are given two unit dual quaternions, η_1 and η_2? The problem is to find the dual quaternion valued function of time, denoted $\operatorname{slerp}(\eta_1, \eta_2, t)$, such that at $t = 0$, it takes the value η_1, that at $t = 1$, it takes the value η_2, and such that its twist, θ, is a constant vector dual quaternion. The name of the function, 'slerp', comes from the equivalent problem for orientations [23].

This requires two functions. The first is the exponential function:

$$\exp(\theta) = \sum_{k=0}^{\infty} \frac{\theta^k}{k!}, \tag{39}$$

The function $\exp(t\theta)$ as function of time t is the pose traveled if it is the identity pose at $t = 0$, and maintains a constant twist θ, that is

$$\frac{d}{dt}\exp(t\theta) = \theta\exp(t\theta) = \exp(t\theta)\theta. \tag{40}$$

The second is a logarithm function, which given a unit dual quaternion η, finds θ such that $\exp(\theta) = \eta$. There are infinitely many possibilities for θ, so we will just describe the principal value, that is, the one for which the norm of the angular velocity is minimized. Then

$$\begin{aligned} \operatorname{slerp}(\eta_1, \eta_2, t) &= \eta_1 \exp(t \log(\eta_1^*\eta_2)) \\ &= \exp(t \log(\eta_2\eta_1^*))\eta_1. \end{aligned} \tag{41}$$

If one is merely interested in interpolating between the poses represented by these unit dual quaternions, multiply one of them by ± 1 so that non-dual part of $\eta_1\eta_2^{-1}$ has non-negative real part.

The results of this section may be found in [16]. We have seen similar results in [22, 27, 28], but we believe our formulas to be more explicit.

Given any dual quaternion

$$\eta = c + \boldsymbol{d} + \epsilon(x + \boldsymbol{y}), \tag{42}$$

by decomposing $\boldsymbol{y}$ into a vector parallel to $\boldsymbol{d}$ and perpendicular to $\boldsymbol{d}$, we can suppose without loss of generality that there exists orthonormal vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ such that η is a linear combination of 1, $\boldsymbol{a}$, ϵ, $\epsilon\boldsymbol{a}$, and $\epsilon\boldsymbol{b}$.

If $\boldsymbol{a}$ and $\boldsymbol{b}$ are orthonormal vectors, then if $w \neq 0$

$$\begin{aligned} &\exp\left(\tfrac{1}{2}wt\boldsymbol{a} + \epsilon(\tfrac{1}{2}v_1t\boldsymbol{a} + \tfrac{1}{2}v_2t\boldsymbol{b})\right) \\ &= \left(\cos(\tfrac{1}{2}wt) + \sin(\tfrac{1}{2}wt)\boldsymbol{a}\right)\left(1 + \tfrac{1}{2}\epsilon v_1t\boldsymbol{a}\right) \\ &\quad + \epsilon\frac{v_2}{w}\sin(\tfrac{1}{2}wt)\boldsymbol{b}, \end{aligned} \tag{43}$$

and if $w = 0$

$$\exp\left(\tfrac{1}{2}\epsilon vt\boldsymbol{a}\right) = 1 + \tfrac{1}{2}\epsilon vt\boldsymbol{a}. \tag{44}$$

If $w \neq 0$, this represents a pose that rotates counterclockwise with angular velocity w around $\boldsymbol{a}$, and translates by

$$\boldsymbol{t} = v_1t\boldsymbol{a} + \frac{v_2}{w}\big(\sin(wt)\boldsymbol{b} + (1 - \cos(wt))(\boldsymbol{a} \times \boldsymbol{b})\big), \tag{45}$$

that is, the sum of a point on a line in the direction of $\boldsymbol{a}$ moving with speed v_1, and a point on a circle in the plane perpendicular to $\boldsymbol{a}$ of radius v_2/w which is traversed once every $2\pi/w$ time units. If $w = 0$, this represents a pose with the identity rotation and translating in a straight line with velocity v.

Suppose $\boldsymbol{a}$ and $\boldsymbol{b}$ are orthonormal vectors, and that

$$\eta = c + s\boldsymbol{a} + \epsilon(x + y_1\boldsymbol{a} + y_2\boldsymbol{b}) \tag{46}$$

is a unit dual quaternion (so $c^2+s^2 = 1$ and $cx+sy_1 = 0$), with $c \neq -1$. Let

$$t = \operatorname{atan2}(s, c), \tag{47}$$

that is, the angle part of the polar coordinates of (c, s), with $-\pi < t \leq \pi$. If $t \neq \pi$ then

$$\log(\eta) = t\boldsymbol{a} + \epsilon(cy_1 - sx)\boldsymbol{a} + \epsilon\frac{ty_2}{s}\boldsymbol{b}, \tag{48}$$

where if $s = 0$ we assume $y_2 = 0$, and we set $y_2/s = 0$.

It is not possible to assign a canonical principal value for $\log(-1)$, because it could be $\pi \boldsymbol{a}$ for any unit vector $\boldsymbol{a}$.

9. Proofs

Proof of Equations (13) *and* (15). First note that any dual number commutes with any dual quaternion. Thus

$$|\eta_1\eta_2|^2 = \eta_2^*\eta_1^*\eta_1\eta_2 = \eta_2^*|\eta_1|^2\eta_2 = |\eta_1|^2\eta_2^*\eta_2 = (|\eta_1||\eta_2|)^2, \tag{49}$$

and

$$\widehat{\eta_1\eta_2} = |\eta_1\eta_2|^{-1}\eta_1\eta_2 = |\eta_1|^{-1}|\eta_2|^{-1}\eta_1\eta_2 = |\eta_1|^{-1}\eta_1|\eta_2|^{-1}\eta_2 = \widehat{\eta}_1\widehat{\eta}_2. \tag{50}$$

□

Proof of Equation (26). First, if we differentiate $\eta^*\eta = 1$, we obtain $\varphi + \varphi^* = 0$, that is, φ is vector.

Suppose the position $\boldsymbol{s}$ is the image of the constant position $\boldsymbol{r}$ under the pose represented by $(\mathsf{R}, \boldsymbol{t})$ and also represented by $\eta = Q + \epsilon B$. Differentiating equation (2) we obtain

$$\dot{\boldsymbol{s}} = \mathsf{R}\mathsf{S}(\boldsymbol{w})\boldsymbol{r} + \mathsf{R}\boldsymbol{v} = \mathsf{R}(\boldsymbol{w} \times \boldsymbol{r} + \boldsymbol{v}). \tag{51}$$

Differentiating equation (18) we obtain

$$\epsilon\dot{\boldsymbol{s}} = \frac{d}{dt}\big(\eta(1 + \epsilon\boldsymbol{r})\overline{\eta}^*\big) = \eta\big(\varphi(1 + \epsilon\boldsymbol{r}) + (1 + \epsilon\boldsymbol{r})\overline{\varphi}^*\big)\overline{\eta}^*. \tag{52}$$

After some manipulation, and setting $\varphi = \frac{1}{2}\boldsymbol{a} + \frac{1}{2}\epsilon\boldsymbol{b}$, we obtain

$$\dot{\boldsymbol{s}} = Q(\boldsymbol{a} \times \boldsymbol{r} + \boldsymbol{b})Q^*. \tag{53}$$

Comparing equations (51) and (53), we obtain

$$\boldsymbol{w} \times \boldsymbol{r} + \boldsymbol{v} = \boldsymbol{a} \times \boldsymbol{r} + \boldsymbol{b}, \tag{54}$$

and since this holds for all $\boldsymbol{r}$, the result follows. □

10. Projection from matrices to orthogonal matrices

What would be a good projection of (3×3) matrices onto rotation matrices? One suggestion is $\mathsf{A} \mapsto \widehat{\mathsf{A}} = (\mathsf{A}\mathsf{A}^T)^{-1/2}\mathsf{A} = \mathsf{A}(\mathsf{A}^T\mathsf{A})^{-1/2}$, which works if we have $\det(\mathsf{A}) > 0$. One way to compute it is via the singular value decomposition $\mathsf{A} = \mathsf{U}\Sigma\mathsf{V}^T$, when $\widehat{\mathsf{A}} = \mathsf{U}\mathsf{V}^T$. However this is computationally expensive. See [11].

References

[1] Bruno Vilhena Adorno, Robot Kinematic Modeling and Control Based on Dual Quaternion Algebra – Part I: Fundamentals, 2017, hal-01478225.

[2] Om Prakash Agrawal. Hamilton operators and dual-number-quaternions in spatial kinematics. Mechanism and Machine Theory, 22(6):569-575, Jan 1987.

[3] S.L. Altmann, Hamilton, Rodrigues, and the Quaternion Scandal, Mathematics Magazine, Vol. 62, No. 5. (Dec., 1989), pp. 291-308.

[4] R.S. Ball, The theory of screws: A study in the dynamics of a rigid body. Hodges, Foster and Co., Dublin, 1876.

[5] O. Bottema, B. Roth, Theoretical Kinematics, North-Holland, Amsterdam, 1979.

[6] Gregory S Chirikjian, Robert Mahony, Sipu Ruan, and Jochen Trumpf. Pose Changes From a Different Point of View. Journal of Mechanisms and Robotics, 10(2):021008-021008-12, Feb 2018.

[7] M.A. Clifford, Preliminary Sketch of Biquaternions, Proceedings of the London Mathematical Society, Volume s1-4, Issue 1, November 1871, Pages 381-395, https://doi.org/10.1112/plms/s1-4.1.381.

[8] J. R. Dooley and J. M. McCarthy, Spatial rigid body dynamics using dual quaternion components, Proceedings. 1991 IEEE International Conference on Robotics and Automation, Sacramento, CA, USA, 1991, pp. 90-95 vol.1, doi: 10.1109/ROBOT.1991.131559.

[9] Jaime Gallardo-Alvarado, Kinematic Analysis of Parallel Manipulators by Algebraic Screw Theory, Springer, Switzerland, 2016.

[10] Da-Peng Han, Qing Wei, and Ze-Xiang Li, Kinematic Control of Free Rigid Bodies Using Dual Quaternions, International Journal of Automation and Computing 05(3), July 2008, 319-324, DOI: 10.1007/s11633-008-0319-1.

[11] N.J. Higham, Matrix nearness problems and applications, in M.J.C. Gover and S. Barnett, editors, Applications of Matrix Theory, pages 1-27. Oxford University Press, 1989.

[12] L. Kavan, S. Collins, J. Žára, C. O'Sullivan, Skinning with Dual Quaternions, https://dl.acm.org/doi/pdf/10.1145/1230100.1230107.

[13] L. Kavan, S. Collins, J. Žára, C. O'Sullivan, Geometric Skinning with Approximate Dual Quaternion Blending, ACM Transactions on Graphics, Vol. 27, No. 4, Article 105, Publication date: October 2008.

[14] Ben Kenwright, A Beginners Guide to Dual-Quaternions, What They Are, How They Work, and How to Use Them for 3D Character Hierarchies, https://cs.gmu.edu/~jmlien/teaching/cs451/uploads/Main/dual-quaternion.pdf.

[15] Hugo T.M. Kussaba, Luis F.C. Figueredo, João Y. Ishihara, and Bruno V. Adorno, Hybrid kinematic control for rigid body pose stabilization using dual quaternions, Journal of the Franklin Institute, 354(7):2769-2787, May 2017.

[16] Stephen Montgomery-Smith, Functional calculus for dual quaternions. Adv. Appl. Clifford Algebras 33, 36 (2023). https://doi.org/10.1007/s00006-023-01282-y.

[17] A. Perez and J.M. McCarthy, Dual Quaternion Synthesis of Constrained Robotic Systems, ASME. J. Mech. Des. May 2004 126(3) 425-435.

[18] J. Pujol, Hamilton, Rodrigues, Gauss, Quaternions, and Rotations: a Historical Reassessment, Commun. Math. Anal., Volume 13, Number 2 (2012), 1-14.

[19] M. Schilling, Universally manipulable body models—dual quaternion representations in layered and dynamic MMCs, Auton Robot 30, 399 (2011), https://doi.org/10.1007/s10514-011-9226-3https://link.springer.com/article/10.1007/s10514-011-9226-3.
[20] M. Schilling, Hierarchical Dual Quaternion-Based Recurrent Neural Network as a Flexible Internal Body Model, 2019 International Joint Conference on Neural Networks (IJCNN), 2019, pp. 1-8, doi: 10.1109/IJCNN.2019.8852328, https://ieeexplore.ieee.org/abstract/document/8852328.
[21] J M Selig, Geometric fundamentals of robotics, Springer-Verlag New York Inc., 2nd edition, 2005.
[22] J.M. Selig, Exponential and Cayley Maps for Dual Quaternions, Advances in Applied Clifford Algebras, 20(3-4):923-936, May 2010.
[23] Ken Shoemake, Animating Rotation with Quaternion Curves, SIGGRAPH 1985, https://www.cs.cmu.edu/~kiranb/animation/p245-shoemake.pdf.
[24] Frederico F. A. Silva, Juan José Quiroz-Omaña, and Bruno V. Adorno, Dynamics of Mobile Manipulators using Dual Quaternion Algebra, Journal of Mechanisms and Robotics, pages 1-24, Apr 2022.
[25] Mark W. Spong, Seth Hutchinson, and M. Vidyasagar, Robot Modeling and Control, Wiley, 2006.
[26] Stanford Artificial Intelligence Laboratory et al., Robotic Operating System, https://www.ros.org, 2018.
[27] Xiangke Wang, Dapeng Han, Changbin Yu, and Zhiqiang Zheng, The geometric structure of unit dual quaternions with application in kinematic control, Journal of Mathematical Analysis and Applications 389(2), 2012, 1352-1364.
[28] Yuanqing Wu, J.M. Selig and Marco Carricato, (2019) Parallel Robots with Homokinetic Joints: The Zero-Torsion Case, In: Uhl T. (eds) "Advances in Mechanism and Machine Science." IFToMM WC 2019. Mechanisms and Machine Science, vol 73. Springer, Cham, pp. 269-278.
[29] Wikipedia, Hodge star operator, https://en.wikipedia.org/wiki/Hodge_star_operator#Three_dimensions.
[30] XiaoLong Yang, HongTao Wu, Yao Li, Bai Chen, A dual quaternion solution to the forward kinematics of a class of six-DOF parallel robots with full or reductant actuation, Mechanism and Machine Theory 107 (2017) 27-36, http://dx.doi.org/10.1016/j.mechmachtheory.2016.08.003.

Department of Mathematics, University of Missouri, Columbia, MO 65211.
Email address: stephen@missouri.edu
URL: https://stephenmontgomerysmith.github.io

Johnson Space Center, 2101 E. NASA Pkwy, Houston, TX 77058.
Email address: cecil.shy-1@nasa.gov