Introduction to quaternions

1. Introduction

Invented and developed by William Hamilton in 1843, *quaternions* are essentially a generalization of complex numbers to four dimensions (one real dimension, three imaginary). Quaternions have important, desirable properties when used to represent rotations, and, as such, are worthy of study in this course, where rotations between coordinate frames will occupy much of our attention, especially early on. This document gives a *brief* introduction on quaternions and follows closely the notation in [1], which offers a much more exhaustive treatment.

2. Basic definition

A quaternion q consists of a scalar part s, $s \in \Re$, and a vector part $\mathbf{v} = (x, y, z)$, $\mathbf{v} \in \Re^3$. We will use several different forms to denote quaternions. These are given in equations (1) through (3) below:

$$q \equiv [s, \mathbf{v}] \tag{1}$$

$$q \equiv [s, (x, y, z)] \tag{2}$$

$$q \equiv s + \mathbf{i}x + \mathbf{j}y + \mathbf{k}z \tag{3}$$

In definition (3) above, the imaginary number $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$ have the following properties:

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = -1 \tag{4}$$

$$\mathbf{ij} = \mathbf{k} \tag{5}$$

$$\mathbf{ji} = -\mathbf{k} \tag{6}$$

From (4) through (6) the following additional properties can be derived:

$$\mathbf{ijk} = -1 \tag{7}$$

$$\mathbf{jk} = \mathbf{i} \tag{8}$$

$$\mathbf{kj} = -\mathbf{i} \tag{9}$$

$$\mathbf{ik} = -\mathbf{j} \tag{10}$$

$$\mathbf{ki} = \mathbf{j} \tag{11}$$

3. Basic properties

A. Addition

Let,

$$q = [s, \mathbf{v}] = [s, (x, y, z)], \tag{12}$$

and,

$$q' = [s', \mathbf{v}'] = [s', (x', y', z')]. \tag{13}$$

Then,

$$\begin{aligned} q + q' &= [s + s', \mathbf{v} + \mathbf{v}'] \\ &= (s + s') + \mathbf{i}(x + x') + \mathbf{j}(y + y') + \mathbf{k}(z + z') \end{aligned} \tag{14}$$

Clearly, quaternion addition is associative and commutative.

B. Multiplication

With definitions (12) and (13), the *product* of two quaternions q and q' is given by,

$$qq' = [ss' - \mathbf{v}\cdot\mathbf{v}', \mathbf{v}\times\mathbf{v}' + s\mathbf{v}' + s'\mathbf{v}] \tag{15}$$

Equation (15) can be verified by writing,

$$\begin{aligned} qq' &\equiv [s, \mathbf{v}][s', \mathbf{v}'] \\ &\equiv (s + \mathbf{i}x + \mathbf{j}y + \mathbf{k}z)(s' + \mathbf{i}x' + \mathbf{j}y' + \mathbf{k}z') \end{aligned} \tag{16}$$

expanding the product in equation (16), applying properties (4) through (11), and grouping terms accordingly. As a reminder, note that for two vectors $\mathbf{a} = (a_x, a_y, a_z)$ and $\mathbf{b} = (b_x, b_y, b_z)$, the dot and cross products are defined as,

$$\mathbf{a}\cdot\mathbf{b} \equiv a_x b_x + a_y b_y + a_z b_z = \mathbf{b}\cdot\mathbf{a} \tag{17}$$

$$\mathbf{a}\times\mathbf{b} \equiv (a_y b_z - a_z b_y, a_z b_x - a_x b_z, a_x b_y - a_y b_x) = -\mathbf{b}\times\mathbf{a} \tag{18}$$

respectively. Note that two purely scalar quaternions, $[s, \mathbf{0}]$ and $[s', \mathbf{0}]$, result in a scalar product, while two purely vector quaternions, $[0, \mathbf{v}]$ and $[0, \mathbf{v}']$, yield both the dot product (albeit negative), and the cross product,

$$[0, \mathbf{v}][0, \mathbf{v}'] = [-\mathbf{v}\cdot\mathbf{v}', \mathbf{v}\times\mathbf{v}']\,. \tag{19}$$

It can be easily verified that multiplication is associative, but *not* commutative; that is, in general, the order of multiplication does matter. One case of multiplication that *is* commutative is multiplication of a general quaternion by a scalar quaternion:

$$[r, \mathbf{0}][s, \mathbf{v}] = [s, \mathbf{v}][r, \mathbf{0}] = [rs, r\mathbf{v}]\,. \tag{20}$$

C. Conjugate

The *conjugate* of a quaternion $q = [s, \mathbf{v}]$ is denoted as q^* and is defined as,

$$q^* = [s, -\mathbf{v}]\,. \tag{21}$$

D. Norm

The norm (or length) of a quaternion q [as defined in equation (12)] is given by,

$$\|q\| = \sqrt{qq^*} \tag{22}$$

Since,

$$\begin{aligned} qq^* &= [s^2 - (\mathbf{v}\cdot(-\mathbf{v})), \mathbf{v}\times(-\mathbf{v}) + s\mathbf{v} + s(-\mathbf{v})] \\ &= [s^2 + \mathbf{v}\cdot\mathbf{v}, \mathbf{0}] \end{aligned} \tag{23}$$

equation (22) can be expanded to,

$$\|q\| = \sqrt{s^2 + x^2 + y^2 + z^2} \tag{24}$$

Note the similarity of equation (24) to the computation of the length of a vector. A *unit quaternion* is any quaternion of unit length such that $\|q\| = 1$.

E. Inversion

The inverse of a quaternion q [as defined in equation (12)] is given by,

$$q^{-1} = q^*/\|q\|^2\,. \tag{25}$$

This can be easily verified by computing qq^{-1}:

$$qq^{-1} = q^{-1}q = qq^*/\|q\|^2 = \|q\|^2/\|q\|^2 = [1, \mathbf{0}]\,. \tag{26}$$

4. Rotations through quaternions

A. Basics

Below we will show that every *unit* quaternion represents a unique rotation in space. It is easy to show that any *unit* quaternion can be expressed as,

$$q = [\cos(\theta/2), \sin(\theta/2)\hat{\mathbf{k}}] \tag{27}$$

where $\hat{\mathbf{k}} = (k_x, k_y, k_z)$ denotes an arbitrary unit vector. Such a unit quaternion represents a rotation of θ about the vector $\hat{\mathbf{k}}$.

Now, let $\mathbf{p} = (p_x, p_y, p_z)$ denote the Cartesian coordinates of a point in space. Let us assume that we wish to rotate $\mathbf{p}$ by θ about the vector $\hat{\mathbf{k}}$ to $\mathbf{p}'$. Let,

$$p = [0, \mathbf{p}]\text{, and,} \tag{28}$$

$$p' = [0, \mathbf{p}']\text{.} \tag{29}$$

Then,

$$p' = qpq^* \tag{30}$$

In words, rotation by an angle θ about a unit vector $\hat{\mathbf{k}}$ can be achieved by pre- and post-multiplying the quaternion representation p of the vector $\mathbf{p}$ by the quaternion q [as given by (27)] and its conjugate q^*, respectively.

Let us show that equation (30) results in the same rotation matrix $R_{\hat{\mathbf{k}}}(\theta)$ as previously derived in class through an argument of composite rotations (as demonstrated in *Mathematica*).

$$p' = qpq^* \tag{31}$$

$$p' = [s, \mathbf{v}][0, \mathbf{p}][s, -\mathbf{v}] \tag{32}$$

$$p' = [-\mathbf{v}\cdot\mathbf{p}, \mathbf{v}\times\mathbf{p} + s\mathbf{p}][s, -\mathbf{v}] \tag{33}$$

$$p' = \begin{matrix}[-s(\mathbf{v}\cdot\mathbf{p}) + (\mathbf{v}\times\mathbf{p})\cdot\mathbf{v} + s(\mathbf{p}\cdot\mathbf{v}), \\ -(\mathbf{v}\times\mathbf{p})\times\mathbf{v} - s(\mathbf{p}\times\mathbf{v}) + s(\mathbf{v}\times\mathbf{p}) + s^2\mathbf{p} + \mathbf{v}(\mathbf{v}\cdot\mathbf{p})]\end{matrix} \tag{34}$$

Now, expression (34) can be significantly simplified. Note that the scalar part of (34) reduces to zero, since the dot products $-s(\mathbf{v}\cdot\mathbf{p})$ and $s(\mathbf{p}\cdot\mathbf{v})$ cancel each other, and,

$$(\mathbf{v}\times\mathbf{p})\cdot\mathbf{v} = 0 \tag{35}$$

which can be verified by inserting definitions (17) and (18) for the dot and cross products, respectively, into (35). For the vector part,

$$-s(\mathbf{p}\times\mathbf{v}) + s(\mathbf{v}\times\mathbf{p}) = 2s(\mathbf{v}\times\mathbf{p}) \tag{36}$$

and,

$$(\mathbf{v}\times\mathbf{p})\times\mathbf{v} = (\mathbf{v}\cdot\mathbf{v})\mathbf{p} - (\mathbf{v}\cdot\mathbf{p})\mathbf{v} \text{ [easily verified through (17) and (18)]} \tag{37}$$

so that,

$$p' = [0, (s^2 - \mathbf{v}\cdot\mathbf{v})\mathbf{p} + 2s(\mathbf{v}\times\mathbf{p}) + 2(\mathbf{v}\cdot\mathbf{p})\mathbf{v}] \tag{38}$$

or, equivalently,

$$\mathbf{p}' = (s^2 - \mathbf{v}\cdot\mathbf{v})\mathbf{p} + 2s(\mathbf{v}\times\mathbf{p}) + 2(\mathbf{v}\cdot\mathbf{p})\mathbf{v}\text{.} \tag{39}$$

We are now ready to substitute (27) into (39):

$$\mathbf{p}' = [\cos^2(\theta/2) - \sin^2(\theta/2)]\mathbf{p} + 2\cos(\theta/2)\sin(\theta/2)\hat{\mathbf{k}}\times\mathbf{p} + 2\sin^2(\theta/2)(\hat{\mathbf{k}}\cdot\mathbf{p})\hat{\mathbf{k}} \tag{40}$$

Applying the following trigonometric half-angle identities,

$$\cos\theta = \cos^2(\theta/2) - \sin^2(\theta/2) \tag{41}$$

$$\sin\theta = 2\cos(\theta/2)\sin(\theta/2) \tag{42}$$

$$(1-\cos\theta) = (1-\cos^2(\theta/2)+\sin^2(\theta/2)) = 2\sin^2(\theta/2) \text{ [from (41) and (42)]} \tag{43}$$

equation (40) simplifies even further to:

$$\mathbf{p}' = (\cos\theta)\mathbf{p} + (\sin\theta)\hat{\mathbf{k}}\times\mathbf{p} + (1-\cos\theta)(\hat{\mathbf{k}}\cdot\mathbf{p})\hat{\mathbf{k}} \tag{44}$$

$$\mathbf{p}' = (c\theta)\mathbf{p} + (s\theta)\hat{\mathbf{k}}\times\mathbf{p} + (v\theta)(\hat{\mathbf{k}}\cdot\mathbf{p})\hat{\mathbf{k}}, \tag{45}$$

where $c\theta = \cos\theta$, $s\theta = \sin\theta$ and $v\theta = 1-\cos\theta$. Equation (44) is an important result known as *Rodrigues formula*. When we expand equation (45) by component terms, we get (after a lot of math best left to *Mathematica*),

$$\mathbf{p}' = R_{\hat{\mathbf{k}}}(\theta)\mathbf{p} \tag{46}$$

$$R_{\hat{\mathbf{k}}}(\theta) = \begin{bmatrix} (k_x^2 v\theta + c\theta) & (k_x k_y v\theta - k_z s\theta) & (k_x k_z v\theta + k_y s\theta) \\ (k_x k_y v\theta + k_z s\theta) & (k_y^2 v\theta + c\theta) & (k_y k_z v\theta - k_x s\theta) \\ (k_x k_z v\theta - k_y s\theta) & (k_y k_z v\theta + k_x s\theta) & (k_y^2 v\theta + c\theta) \end{bmatrix} \tag{47}$$

which is exactly the same result as obtained in class through an argument of composite rotations. The two most important equations in this section are (31) and (45), which are reprinted below to emphasize their importance:

$$p' = qpq^* \text{ [rotation through quaternions]} \tag{48}$$

$$\mathbf{p}' = (c\theta)\mathbf{p} + (s\theta)\hat{\mathbf{k}}\times\mathbf{p} + (v\theta)(\hat{\mathbf{k}}\cdot\mathbf{p})\hat{\mathbf{k}} \text{ [Rodrigues formula]} \tag{49}$$

B. Composite quaternion rotations

Assume two rotations: q_o followed by q_1. The result p' of the first rotation is given by,

$$p' = q_0 p q_0^* \tag{50}$$

while the result p'' after both rotations is given by,

$$p'' = q_1 p' q_1^* \tag{51}$$

$$p'' = q_1(q_0 p q_0^*)q_1^* \tag{52}$$

$$p'' = (q_1 q_0)p(q_0^* q_1^*) \tag{53}$$

$$p'' = (q_1 q_0)p(q_1 q_0)^* \tag{54}$$

Note that from (53) to (54) we used the following fact:

$$q_0^* q_1^* = (q_1 q_0)^* \tag{55}$$

for unit quaternions, which can be readily shown through the definition of quaternion multiplication. Equation (55) is important in that it shows that composite rotations can be computed through simple unit quaternion multiplication.

C. Quaternion to axis-angle representation

For any quaternion q,

$$q = [s, \mathbf{v}] \tag{56}$$

we can readily compute the equivalent axis-angle representation. From (27),

$$q = [\cos(\theta/2), \sin(\theta/2)\hat{\mathbf{k}}] \tag{57}$$

we can compute θ and $\hat{\mathbf{k}}$ as:

$$\theta = 2\,\mathrm{atan}(\|\mathbf{v}\|, s) \tag{58}$$

$$\hat{\mathbf{k}} = \mathbf{v}/\|\mathbf{v}\| \tag{59}$$

Note that when $\theta = 0$, the axis of rotation becomes ill-conditioned, since *any* axis is equivalent for a null rotation.

D. Quaternion to rotation matrix

Suppose we have a quaternion q,

$$q = [s, (x, y, z)]\,. \tag{60}$$

and would like to identify the corresponding 3×3 rotation matrix R. We start by expanding equation (39) into the components of $\mathbf{v} = (x, y, z)$:

$$\mathbf{p}' = (s^2 - \mathbf{v} \cdot \mathbf{v})\mathbf{p} + 2s(\mathbf{v} \times \mathbf{p}) + 2(\mathbf{v} \cdot \mathbf{p})\mathbf{v} \tag{61}$$

$$\mathbf{p}' = \left((s^2 - x^2 - y^2 - z^2)\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + 2s\begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} + 2\begin{bmatrix} x^2 & xy & xz \\ xy & y^2 & yz \\ xz & yz & z^2 \end{bmatrix} \right)\mathbf{p} \tag{62}$$

$$\mathbf{p}' = \begin{bmatrix} 1 - 2(y^2 + z^2) & 2(xy - sz) & 2(xz + sy) \\ 2(xy + sz) & 1 - 2(x^2 + z^2) & 2(yz - sx) \\ 2(xz - sy) & 2(yz + sx) & 1 - 2(x^2 + y^2) \end{bmatrix}\mathbf{p} \tag{63}$$

Note that in (62), we used the following matrix representation of the cross product:

$$(\mathbf{v} \times \mathbf{p}) = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}\mathbf{p} \tag{64}$$

which is easily verified by doing a component-by-component comparison of both sides of equation (64). Thus, R is given by,

$$R = \begin{bmatrix} 1 - 2(y^2 + z^2) & 2(xy - sz) & 2(xz + sy) \\ 2(xy + sz) & 1 - 2(x^2 + z^2) & 2(yz - sx) \\ 2(xz - sy) & 2(yz + sx) & 1 - 2(x^2 + y^2) \end{bmatrix} \tag{65}$$

E. Rotation matrix to quaternion

Suppose we have a rotation matrix R,

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{66}$$

such that,

$$\mathbf{p}' = R\mathbf{p}, \tag{67}$$

and would like to identify the corresponding quaternion q,

$$q = [s, (x, y, z)]. \tag{68}$$

From the result of the previous section in (65), we can write down the following 10 convenient equations relating r_{ij} to $[s, (x, y, z)]$:

$$s^2 = \frac{1}{4}(1 + r_{11} + r_{22} + r_{33}) \tag{69}$$

$$x^2 = \frac{1}{4}(1 + r_{11} - r_{22} - r_{33}) \tag{70}$$

$$y^2 = \frac{1}{4}(1 - r_{11} + r_{22} - r_{33}) \tag{71}$$

$$z^2 = \frac{1}{4}(1 - r_{11} - r_{22} + r_{33}) \tag{72}$$

and,

$$sx = \frac{1}{4}(r_{32} - r_{23}) \tag{73}$$

$$sy = \frac{1}{4}(r_{13} - r_{31}) \tag{74}$$

$$sz = \frac{1}{4}(r_{21} - r_{12}) \tag{75}$$

$$xy = \frac{1}{4}(r_{12} + r_{21}) \tag{76}$$

$$xz = \frac{1}{4}(r_{13} + r_{31}) \tag{77}$$

$$yz = \frac{1}{4}(r_{23} + r_{32}) \tag{78}$$

Now, to solve for the parameters $[s, (x, y, z)]$ robustly, use the first four equations [(69) through (72)], to solve for the largest valued parameter in $\{s^2, x^2, y^2, z^2\}$. Then, solve for the remaining three parameters using three of the six remaining equations [(73) through (78)].

5. Rotation trajectories through quaternions

A. Linear interpolation (Lerp)

Suppose you wish to generate a trajectory of orientations between two rotations represented by unit quaternions q_0 and q_1 respectively. One method of achieving this is called Linear Interpolation (Lerp), which is given by,

$$\text{Lerp}(q_0, q_1, h) = \frac{f(q_0, q_1, h)}{\|f(q_0, q_1, h)\|}, \; h \in [0, 1], \tag{79}$$

where,

$$f(q_0, q_1, h) = q_0(1-h) + q_1 h . \tag{80}$$

Note that the normalization in equation (79) is required since $f(q_0, q_1, h)$ is not guaranteed to be a unit vector. It is easy to verify that,

$$\text{Lerp}(q_0, q_1, 0) = q_0 \text{, and,} \tag{81}$$

$$\text{Lerp}(q_0, q_1, 1) = q_1 . \tag{82}$$

See [1] for much more detail.

B. Spherical linear interpolation (Slerp)

One of the main objections to the Lerp trajectory is that the resulting trajectory is not constant velocity. Another method, called Spherical Linear Interpolation (Slerp), achieves the shortest possible interpolation path between q_0 and q_1 (or q_0 and $-q_1$, whichever is shorter) along the four-dimensional unit sphere at constant velocity. The Slerp interpolation is given by,

$$\text{Slerp}(q_0, q_1, h) = (q_1 q_0^{-1})^h q_0, \; h \in [0, 1], \tag{83}$$

or equivalently as,

$$\text{Slerp}(q_0, q_1, h) = (q_1 q_0{}^*)^h q_0, \; h \in [0, 1]. \tag{84}$$

Note that as was the case for Lerp,

$$\text{Slerp}(q_0, q_1, 0) = (q_1 q_0^{-1})^0 q_0 = 1 \cdot q_0 = q_0 \text{, and,} \tag{85}$$

$$\text{Slerp}(q_0, q_1, 1) = (q_1 q_0^{-1})^1 q_0 = q_1 q_0^{-1} q_0 = q_1 . \tag{86}$$

The main difference between the Lerp and the Slerp interpolation is that the Lerp interpolation cuts through the unit quaternion sphere (before being projected back onto it through normalization), while the Slerp interpolation moves at constant velocity on the surface on the unit quaternion sphere. (see Figure 6.4 on page 42 in [1]).

In order to compute (83) [or (84)] for $h \in (0, 1)$, we need a little more quaternion math. First, given a unit quaternion of the form,

$$q = [\cos\Omega, \sin\Omega\hat{\mathbf{k}}] \tag{87}$$

where $\hat{\mathbf{k}}$ is a unit vector, we define,

$$\log(q) \equiv [0, \Omega\hat{\mathbf{k}}] . \tag{88}$$

Second, for a quaternion of the form,

$$q = [0, \Omega\hat{\mathbf{k}}] \tag{89}$$

we define,

$$\exp(q) \equiv [\cos\Omega, \sin\Omega\hat{\mathbf{k}}] . \tag{90}$$

Now, we define,

$$q^h \equiv \exp[h \cdot \log(q)] \tag{91}$$

These definitions have some familiar (with respect to scalars) properties, such as:

$$\log([1, 0]) = [0, 0], \tag{92}$$

$$q^a q^b = q^{a+b}, \tag{93}$$

$$q^{ab} = (q^a)^b. \tag{94}$$

A word of caution, though; not all properties scale from scalars to quaternions; for example,

$$\log(pq) \neq \log(p) + \log(q) \tag{95}$$

for quaternions p, q. Now, we are ready to give a formula for computing the Slerp interpolation. Let,

$$q_0 = [s_0, \mathbf{v}_0] \text{, and,} \tag{96}$$

$$q_1 = [s_1, \mathbf{v}_1]. \tag{97}$$

so that,

$$q_1 q_0^* = [s_1 s_0 + (\mathbf{v}_1 \cdot \mathbf{v}_0), -\mathbf{v}_1 \times \mathbf{v}_0 + s_0 \mathbf{v}_1 - s_1 \mathbf{v}_0]. \tag{98}$$

If we let,

$$q_1 q_0^* = [\cos\Omega, \sin\Omega\hat{\mathbf{k}}] \text{ [as in assumption (87)],} \tag{99}$$

then,

$$\cos\Omega = q_0 \cdot q_1 \equiv s_1 s_0 + (\mathbf{v}_1 \cdot \mathbf{v}_0) \tag{100}$$

$$\Omega = \operatorname{acos}(q_0 \cdot q_1) \text{, and,} \tag{101}$$

$$\mathbf{k} = \mathbf{v}_0 \times \mathbf{v}_1 + s_0 \mathbf{v}_1 - s_1 \mathbf{v}_0 \text{, and,} \tag{102}$$

$$\hat{\mathbf{k}} = \begin{cases} \mathbf{k}/\|\mathbf{k}\| & \mathbf{k} \neq \mathbf{0} \\ \mathbf{0} & \mathbf{k} = \mathbf{0} \end{cases} \tag{103}$$

Finally [from definition (91)],

$$(q_1 q_0^*)^h = \exp[h \cdot \log(q_1 q_0^*)] \tag{104}$$

$$(q_1 q_0^*)^h = \exp([0, h\Omega\hat{\mathbf{k}}]) \tag{105}$$

$$(q_1 q_0^*)^h = [\cos(h\Omega), \sin(h\Omega)\hat{\mathbf{k}}]. \tag{106}$$

Equation (106), together with (100) through (103), allows computation of $\text{Slerp}(q_0, q_1, h) = (q_1 q_0^*)^h q_0$, $h \in [0, 1]$. From (106), note that $(q_1 q_0^*)^0 = [1, \mathbf{0}]$ and $(q_1 q_0^*)^1 = q_1 q_0^*$. For much greater detail, see [1].

6. Conclusions

Unit quaternions represent rotations. They have several advantages over three-angle representations (both fixed axis and Euler angles) and rotation matrices:

1. Unit quaternions do not suffer from singularities, as three-angle conventions do.
2. Unit quaternions represent the most compact way of representing rotations without redundancy or singularity (unlike rotation matrices themselves). One can visualize unit quaternions populating the surface of a sphere in four-dimensional Euclidean space, where each quaternion q and its antipode $-q$ represent a unique rotation in three dimensions.

3. The unit quaternion representation allows us to define and compute a distance metric between two rotations. This is not, in general possible or easy with three-angle conventions or rotation matrices.
4. In finite-precision computations, ensuring proper rotations through frequent re-normalization is easy for unit quaternions, but significantly trickier for rotation matrices.
5. For simulation purposes, it is easy to generate a uniformly random distribution of rotations in quaternion space.
6. Unit quaternions allow for the computation of shortest-path, smooth, continuous-velocity trajectories between two rotations (represented as unit quaternions). Three-angle conventions make this very difficult, if not impossible.

References

[1] E. B. Dam, M. Koch and M. Lillholm, "Quaternions, Interpolation and Animation," DIKU-TR-98/5, Technical Report, Department of Computer Science, University of Copenhagen, 1998 (http://www.diku.dk/research/published/98-5.ps.gz).