Chapter 2
Math Fundamentals

Part 5

2.8 Quaternions

Outline

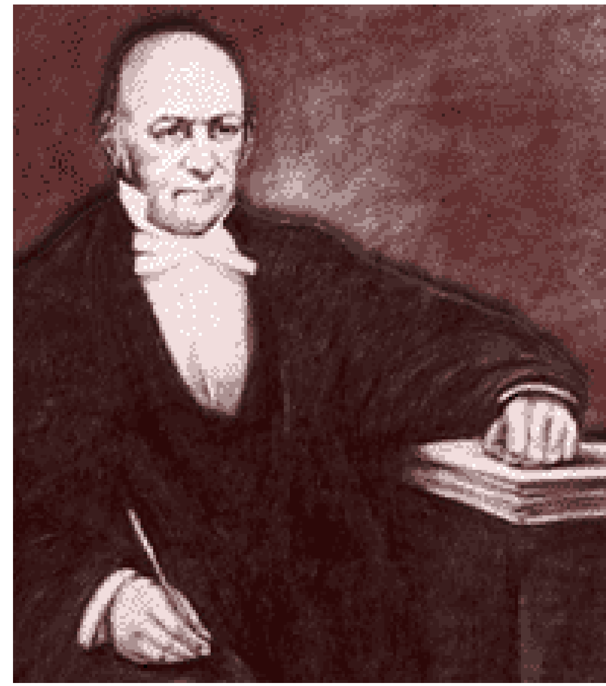
- 2.8.1 Representations and Notation
- 2.7.2 Quaternion Multiplication
- 2.7.3 Other Quaternion Operations
- 2.7.4 Representing 3D Rotations
- 2.7.5 Attitude and Angular Velocity
- Summary

Outline

- 2.8.1 Representations and Notation
- 2.7.2 Quaternion Multiplication
- 2.7.3 Other Quaternion Operations
- 2.7.4 Representing 3D Rotations
- 2.7.5 Attitude and Angular Velocity
- Summary

Smart Irishman: Hamilton

- Quaternions
  - Probably the most powerful number system in common use.
- Hamiltonian mechanics
  - Generalization of Lagrange Mechanics
  - Which was a generalization of Newton-Euler Mechanics.
  - Which was a generalization of Newtonian Mechanics.



Was It All the Guinness?

Core Problem and Properties

- How can we divide a vector by a vector?
- Answer. Need to have "principle imaginaries":

$$i^2 = j^2 = k^2 = ijk = -1$$

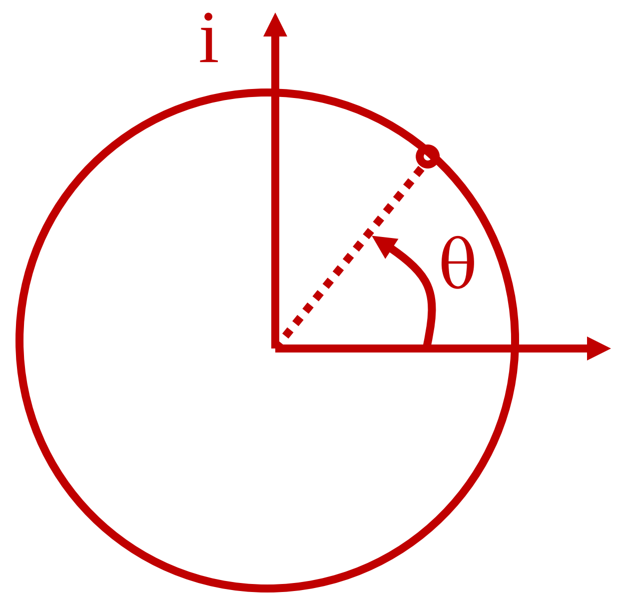
- Quaternions:
  - are generalizations of complex numbers which do not commute (complex #s do).
  - can represent every transformation that an HT can represent.

Why Use Em?

- Only way to solve some problems
  - like the problem of generating regularly spaced 3D angles.
- Best way to solve some problems.
  - No “gimbal lock” at Euler angle singularity.
  - Still not a unique representation though.
- Simplest way to solve some problems.
  - Some problems in registration can be solved in closed form.
- Fastest way to solve some problems.
  - “Quaternion loop” in an inertial navigation system updates vehicle attitude 1000 times a second.

Intuition from Complex Numbers

- Use a second "imaginary" dimension.
- Permits manipulation of rotations like a vector.
  - Remember "phasors" in EE.



Notations

- 4-tuples (q_0, q_1, q_2, q_3)
- Hypercomplex numbers $q = q_0 + q_1 i + q_2 j + q_3 k$
- Sum of real and imaginary parts $\tilde{q} = q + \vec{q}$
- Ordered doublet $(q, \vec{q})$
- Exponential $q = e^{\frac{1}{2}\theta \vec{w}}$

Manipulate like Polynomials

I will use these two

My Preference

- Mostly use the scalar-vector sum form:

$$\tilde{q} = q + \vec{q}$$

~ means quaternion
⇀ means 3D normal vector
means scalar

- Occasionally write it out to get hypercomplex form:

$$q = q_0 + q_1 i + q_2 j + q_3 k$$

Outline

- 2.8.1 Representations and Notation
- <u>2.7.2 Quaternion Multiplication</u>
- 2.7.3 Other Quaternion Operations
- 2.7.4 Representing 3D Rotations
- 2.7.5 Attitude and Angular Velocity
- Summary

Multiplication

- Quaternions are elements of a vector space endowed with multiplication.
  - Just Like Complex Numbers
- The expression:

$$\tilde{p}\tilde{q} = (p_0 + p_1 i + p_2 j + p_3 k)(q_0 + q_1 i + q_2 j + q_3 k)$$

- Gives the sum of all these elements:

	q_0	$q_1 i$	$q_2 j$	$q_3 k$
p_0	$p_0 q_0$	$p_0 q_1 i$	$p_0 q_2 j$	$p_0 q_3 k$
$p_1 i$	$p_1 q_0 i$	$p_1 q_1 i^2$	$p_1 q_2 ij$	$p_1 q_3 ik$
$p_2 j$	$p_2 q_0 j$	$p_2 q_1 ji$	$p_2 q_2 j^2$	$p_2 q_3 jk$
$p_3 k$	$p_3 q_0 k$	$p_3 q_1 ki$	$p_3 q_2 kj$	$p_3 q_3 k^2$

- So, we need to define what i*i etc mean…

Multiplication Rule

- Two goals:
  - 1) Manipulate like polynomials
  - 2) Product of two quaternions is a quaternion.
- To get things to work as Hamilton intended we need to have:

Diagonals work like complex numbers. Off diagonals work like vector cross product.

	i	j	k
i	-1	k	-j
j	-k	-1	i
k	j	-i	-1

- Or, more compactly:

$$i^2 = j^2 = k^2 = ijk = -1$$

Product

- In hypercomplex (polynomial) form:

$$\tilde{p}\tilde{q} = (p_0 + p_1 i + p_2 j + p_3 k)(q_0 + q_1 i + p_2 j + p_3 k)$$

$$\tilde{p}\tilde{q} = (p_0 q_0 - p_1 q_1 - p_2 q_2 - p_3 q_3) + (\ldots) i + \ldots$$

- In vector form:

$$\tilde{p}\tilde{q} = (p + \vec{p})(q + \vec{q})$$

$$\tilde{p}\tilde{q} = pq + p\vec{q} + q\vec{p} + \vec{p}\vec{q} \quad ?$$

- The last term can be written in terms of familiar vector products.

$$\vec{p}\vec{q} = \vec{p} \times \vec{q} - \vec{p} \bullet \vec{q}$$

2 common vector products

- Convenient to summarize like so:

$$\tilde{p}\tilde{q} = pq - \vec{p} \bullet \vec{q} + p\vec{q} + q\vec{p} + \vec{p} \times \vec{q}$$

Not the same thing

Non-Commutativity

- The vector cross product does not commute. Therefore:

$$\tilde{p}\tilde{q} \neq \tilde{q}\tilde{p}$$

What is the source of this property?

15

Outline

- 2.8.1 Representations and Notation
- 2.7.2 Quaternion Multiplication
- <u>2.7.3 Other Quaternion Operations</u>
- 2.7.4 Representing 3D Rotations
- 2.7.5 Attitude and Angular Velocity
- Summary

Addition

- Works just like vectors, polynomials, and complex numbers….

$$\tilde{p} + \tilde{q} = (p_0 + q_0) + (p_1 + q_1)i + (p_2 + q_2)j + (p_3 + q_3)k$$

Distributivity

- Works just like vectors, polynomials, and complex numbers….

$$(\tilde{p} + \tilde{q})\tilde{r} = \tilde{p}\tilde{r} + \tilde{q}\tilde{r}$$

$$\tilde{p}(\tilde{q} + \tilde{r}) = \tilde{p}\tilde{q} + \tilde{p}\tilde{r}$$

Dot Product and Norm

- Works just like vectors, polynomials, and complex numbers....

$$\tilde{p} \bullet \tilde{q} = pq + \vec{p} \bullet \vec{q}$$

Not the same thing

- Can now define a length (norm):

$$|\tilde{q}| = \sqrt{\tilde{q} \bullet \tilde{q}}$$

- <u>Unit quaternions</u> have a norm of unity.

Conjugate

- Works just like complex numbers....

$$\tilde{q}^* = q - \vec{q}$$

- Product with conjugate equals dot product:

$$\tilde{q}\tilde{q}^* = (qq + \vec{q} \bullet \vec{q}) = \tilde{q} \bullet \tilde{q}$$

- Another way to get the norm is then:

$$|\tilde{q}| = \sqrt{\tilde{q}\tilde{q}^*}$$

20

Quaternion Inverse

- The Big Kahuna. Since we have:

$$\tilde{q}\tilde{q}^{*}/|\tilde{q}|^{2} = 1$$

- By definition of inverse:

$$\tilde{q}^{-1} = \tilde{q}^{*}/|\tilde{q}|^{2}$$

- So….

$$\frac{\tilde{p}}{\tilde{q}} = \tilde{p}\tilde{q}^{-1} = \frac{\tilde{p}\tilde{q}^{*}}{|\tilde{q}|^{2}}$$

- That's how you divide a vector by a vector!

Outline

- 2.8.1 Representations and Notation
- 2.7.2 Quaternion Multiplication
- 2.7.3 Other Quaternion Operations
- <u>2.7.4 Representing 3D Rotations</u>
- 2.7.5 Attitude and Angular Velocity
- Summary

Vectors as Quaternions

- "Quaternionize":

$$\tilde{x} = 0 + \vec{x}$$

23

Rotations as Quaternions

- The unit quaternion: $\tilde{q} = \cos\frac{\theta}{2} + \hat{w}\sin\frac{\theta}{2}$

- Represents the operator which rotates by the angle θ around the axis whose unit vector is $\hat{w}$.

- The inverse is clearly: $\hat{w} = \vec{q} / |\vec{q}|$

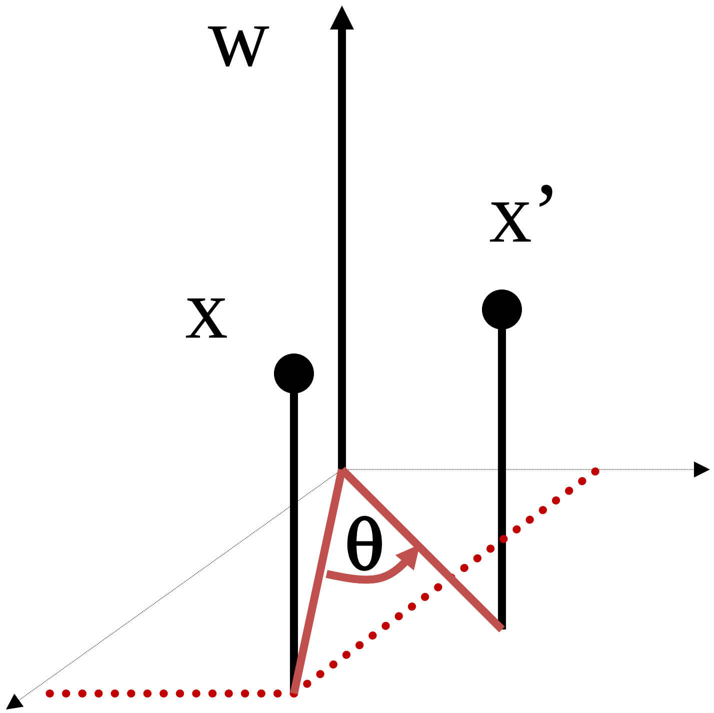
$$\theta = 2\,\mathrm{atan2}(|\vec{q}|, q)$$

Real vectors are just quaternions 0+xi+yj+zk

Rotating a Vector (Point)

- Use the quaternion sandwich:

$$\tilde{x}' = \tilde{q}\tilde{x}\tilde{q}^*$$



Composite Rotations

- Use the composite quaternion sandwich….

- Recall: $\tilde{x}' = \tilde{q}\tilde{x}\tilde{q}^*$

- Thus:

$$\tilde{x}'' = \tilde{p}\tilde{x}'\tilde{p}^* = (\tilde{p}\tilde{q})\tilde{x}(\tilde{q}^*\tilde{p}^*)$$

Conjugate of a product works like matrices!

Composition of operations equals multiplication.

Quaternion to Rot() Matrix

- For the quaternion:

$$q = q_0 + q_1 i + q_2 j + q_3 k$$

- The equivalent Rot() matrix is:

$$R = \begin{bmatrix} 2[q_0^2 + q_1^2] - 1 & 2[q_1 q_2 - q_0 q_3] & 2[q_1 q_3 + q_0 q_2] \\ 2[q_1 q_2 + q_0 q_3] & 2[q_0^2 + q_2^2] - 1 & 2[q_2 q_3 - q_0 q_1] \\ 2[q_1 q_3 - q_0 q_2] & 2[q_0 q_1 + q_2 q_3] & 2[q_0^2 + q_3^2] - 1 \end{bmatrix}$$

Rot() Matrix to Quaternion

- For the Rot() matrix:

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

- The equivalent quaternion is determined from:

$$r_{11} + r_{22} + r_{33} = 4q_0^2 - 1$$
$$r_{11} - r_{22} - r_{33} = 4q_1^2 - 1$$
$$-r_{11} + r_{22} - r_{33} = 4q_2^2 - 1$$
$$-r_{11} - r_{22} + r_{33} = 4q_3^2 - 1$$

Are quaternions unique for a given rotation?

Calculus (wrt scalars)

- Derivatives work like you would expect:

$$\frac{d\tilde{q}}{dt} = \frac{dq}{dt} + \frac{d\vec{q}}{dt}$$

- Integrals work like you would expect:

$$\int_0^t \tilde{q}\, dt = \int_0^t q\, dt + \int_0^t \vec{q}\, dt$$

Outline

- 2.8.1 Representations and Notation
- 2.7.2 Quaternion Multiplication
- 2.7.3 Other Quaternion Operations
- 2.7.4 Representing 3D Rotations
- <u>2.7.5 Attitude and Angular Velocity</u>
- Summary

Angular Velocity

- Define the angular velocity:

$$\tilde{\omega}_n(t) = (\omega_0 + \omega_1 i + \omega_2 j + \omega_3 k)$$

- For a unit quaternion representation of orientation:

$$\tilde{q}(t) = \cos\frac{\theta(t)}{2} + \hat{w}\sin\frac{\theta(t)}{2}$$

- Its time derivative is:

$$\frac{d\tilde{q}(t)}{dt} = \frac{1}{2}\tilde{\omega}_n(t)\tilde{q}(t)$$

Recall the skew matrix derivative of a rotation matrix.

dubyaQ

31

Angular Velocity

- That was for angular velocity represented in navigation coordinates:

$$\tilde{\omega}_n(t) = (\omega_0 + \omega_1 i + \omega_2 j + \omega_3 k)$$

- If you have it in body coordinates, just use the instantaneous value of $\tilde{q}(t)$ itself to convert:

- Substituting: $\tilde{\omega}_n(t) = \tilde{q}(t)\tilde{\omega}_b(t)\tilde{q}(t)^*$

Qdubya

$$\frac{d\tilde{q}(t)}{dt} = \frac{1}{2}\tilde{q}(t)\tilde{\omega}_b(t)$$

"Quaternion Loop"

- Runs at 10 kHz inside an INS:

$$\tilde{q}(t) = \frac{1}{2}\int_0^t \tilde{q}(t)\tilde{\omega}_b(t)\,dt$$

- 16 * 2 = 32 flops

"Quaternion Loop"

- For highest accuracy, we can use Jordan's trick:

$$\tilde{q}_{k+1}^{n} = \tilde{q}_{k+1}^{k}\tilde{q}_{k}^{n} = \frac{1}{2}\int_{t_k}^{t_{k+1}} \tilde{q}_{k}^{n}\tilde{\omega}_k dt = \frac{1}{2}\int_{t_k}^{t_{k+1}} \left({}^{\times}\tilde{\omega}_b\right) dt\ \tilde{q}_{k}^{n} = exp\left\{{}^{\times}[\delta\tilde{\Theta}]\right\}\tilde{q}_{k}^{n}$$

- Define the skew matrix of a quaternion:

$${}^{\times}[\delta\tilde{\Theta}] = \frac{1}{2}\left({}^{\times}\tilde{\omega}_b\right)dt = \frac{1}{2}\begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix} dt$$

"Quaternion Loop"

- But such matrices have closed form exponentials:

$$\tilde{q}_{k+1}^{k} = exp\left\{ {}^{\times}[\delta\tilde{\Theta}] \right\} = I + f_1(\delta\Theta)\ {}^{\times}[\delta\tilde{\Theta}] + f_2(\delta\Theta)\left({}^{\times}[\delta\tilde{\Theta}] \right)^2$$

- Where: $f_1(\delta\Theta) = \dfrac{sin\,\delta\Theta}{\delta\Theta}$ $\quad f_2(\delta\Theta) = \dfrac{(1 - cos\,\delta\Theta)}{\delta\Theta^2}$

- After more manipulation

$$\tilde{q}_{k+1}^{k} = cos\,\delta\Theta[I] + sin\,\delta\Theta\left[\left({}^{\times}[\tilde{\omega}_b] \right) / \left|\vec{\omega}_b\right| \right]$$

35

Outline

- 2.8.1 Representations and Notation
- 2.7.2 Quaternion Multiplication
- 2.7.3 Other Quaternion Operations
- 2.7.4 Representing 3D Rotations
- 2.7.5 Attitude and Angular Velocity
- Summary

Summary

- Quaternions are hypercomplex numbers with an i,j,and k that act like the i in complex numbers.
- Notation is half the battle.
- Provide elegant and efficient ways to model 3D transformations of points (and hence 3D coordinate system conversions).