Visualizing Quaternions

Andrew J. Hanson
Computer Science Department
Indiana University

Siggraph 2005 Tutorial

OUTLINE

I: (45min) **Twisting Belts, Rolling Balls, and Locking Gimbals:**
Explaining Rotation Sequences with Quaternions

II: (45 min) **Quaternion Fields:**
Curves, Surfaces, and Volumes

Part I

Twisting Belts, Rolling Balls, and Locking Gimbals

Explaining Rotation Sequences with Quaternions

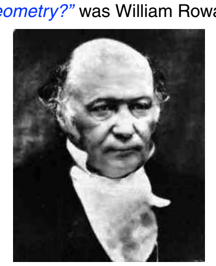
Where Did Quaternions Come From?

...from the discovery of *Complex Numbers:*

- $\boxed{z = x + iy}$ Complex numbers = realization that $z^2 + 1 = 0$ cannot be solved unless you have an "imaginary" number with $i^2 = -1$.
- **Euler's formula**: $e^{i\theta} = \cos\theta + i \sin\theta$ allows you to do most of 2D geometry.

Hamilton

The first to ask *"If you can do 2D geometry with complex numbers, how might you do 3D geometry?"* was William Rowan Hamilton, circa 1840.



Sir William Rowan Hamilton
4 August 1805 — 2 September 1865

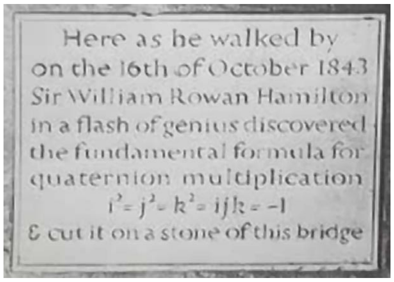
Hamilton's epiphany: 16 October 1843

"An electric circuit seemed to close; and a spark flashed forth ... Nor could I resist the impulse – unphilosophical as it may have been – to cut with a knife on a stone of Brougham Bridge, as we passed it, the fundamental formula with the symbols, i, j, k; namely,

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

which contains the Solution of the Problem..."

...at the site of Hamilton’s carving



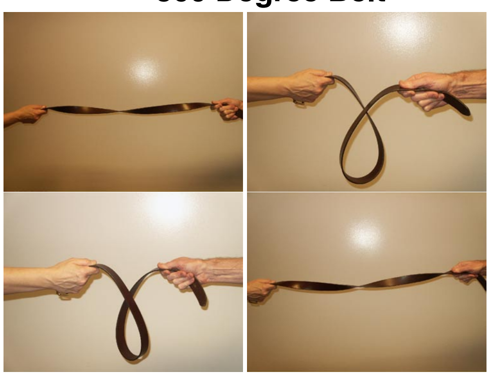
The plaque on Broome Bridge in Dublin, Ireland, commemorating the legendary location where Hamilton conceived of the idea of quaternions. (Hamilton apparently misspelled it as “Brougham Bridge” in his letter.)

The Belt Trick

Quaternion Geometry in our daily lives

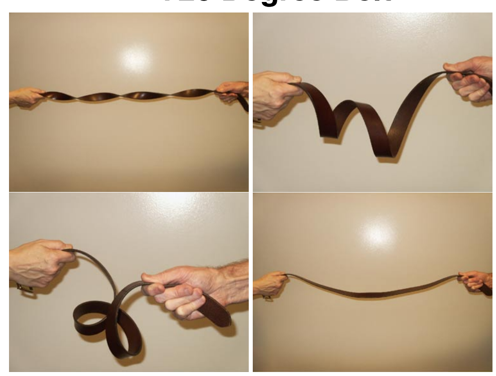
- Two people hold ends of a belt.
- Twist the belt either 360 degrees or 720 degrees.
- **Rule:** *Move belt ends any way you like but do not change orientation of either end.*
- Try to straighten out the belt.

360 Degree Belt



360 twist: stays twisted, can change DIRECTION!

720 Degree Belt

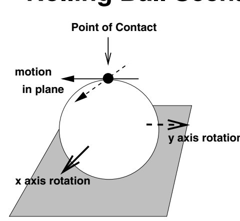


720 twist: CAN FLATTEN OUT WHOLE BELT!

Rolling Ball Puzzle

1. Put a ball on a flat table.
2. Place hand flat on top of the ball
3. Make circular rubbing motion, as though polishing the tabletop.
4. Watch a point on the equator of the ball.
5. *small clockwise circles* → **equator goes counterclockwise**
6. *small counterclockwise circles* → **equator goes clockwise**

Rolling Ball Scenario

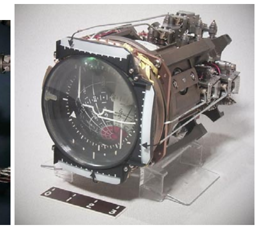


Gimbal Lock

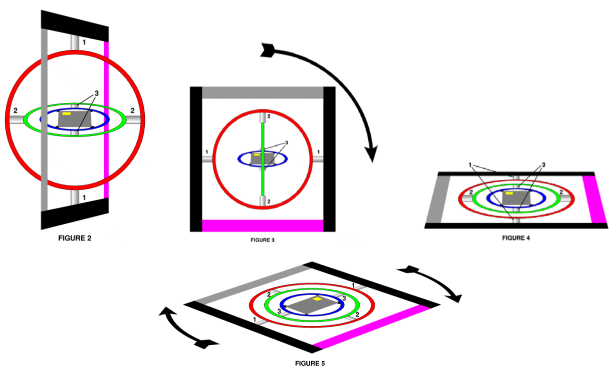
Gimbal Lock occurs when a mechanical or computer system experiences an anomaly due to an (x, y, z)-based orientation control sequence.

- *Mechanical systems cannot avoid all possible gimbal lock situations .*
- *Computer orientation interpolation systems can avoid gimbal-lock-related glitches* **by using quaternion interpolation.**

Gimbal Lock — Apollo Systems



Red-painted area = Danger of real Gimbal Lock



Mechanical Gimbal Lock: Using x, y, z axes to encode orientation gives singular situations.

2D Rotations

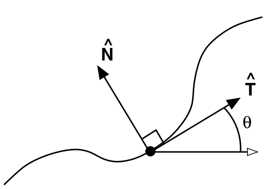
- 2D rotations $\leftrightarrow$ *complex numbers.*
- Why? $e^{i\theta}(x + iy) = (x' + iy')$

$$
\begin{aligned}
x' &= x\cos\theta - y\sin\theta \\
y' &= x\sin\theta + y\cos\theta
\end{aligned}
$$

- **Complex numbers** are a subspace of quaternions — so exploit 2D rotations to **introduce us to quaternions** and their geometric meaning.

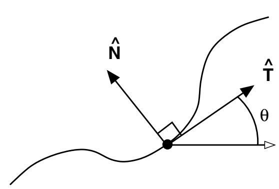
Frames in 2D

The tangent and normal to 2D curve move continuously along the curve:



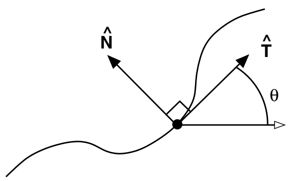
Frames in 2D

The tangent and normal to 2D curve move continuously along the curve:



Frames in 2D

The tangent and normal to 2D curve move continuously along the curve:



19

Frame Matrix in 2D

This motion is described at each point (or time) by the matrix:

$$\begin{aligned} R_2(\theta) &= \begin{bmatrix} \hat{\mathbf{T}} & \hat{\mathbf{N}} \end{bmatrix} \\ &= \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} . \end{aligned}$$

20

The Belt Trick Says:

There is a Problem...at least in 3D

How do you get $\cos\theta$ to know about 720 degrees?

21

The Belt Trick Says:

There is a Problem...at least in 3D

How do you get $\cos\theta$ to know about 720 degrees?

Hmmmmm. $\cos(\theta/2)$ knows about 720 degrees, right?

22

Half-Angle Transform:

A Fix for the Problem?

Let $a = \cos(\theta/2)$, $b = \sin(\theta/2)$,
(i.e., $\cos\theta = a^2 - b^2$, $\sin\theta = 2ab$),
and parameterize 2D rotations as:

$$R_2(a, b) = \begin{bmatrix} a^2 - b^2 & -2ab \\ 2ab & a^2 - b^2 \end{bmatrix} .$$

where orthonormality implies

$$(a^2 + b^2)^2 = 1$$

which reduces back to $a^2 + b^2 = 1$.

23

Frame Evolution in 2D

Examine time-evolution of 2D frame (on our way to 3D): First in $\theta(t)$ coordinates:

$$\begin{bmatrix} \hat{\mathbf{T}} & \hat{\mathbf{N}} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} .$$

Differentiate to find frame equations:

$$\begin{aligned} \dot{\hat{\mathbf{T}}}(t) &= +\kappa\hat{\mathbf{N}} \\ \dot{\hat{\mathbf{N}}}(t) &= -\kappa\hat{\mathbf{T}} , \end{aligned}$$

where $\kappa(t) = d\theta/dt$ is the **curvature**.

24

Frame Evolution in (a, b):

The basis $(\hat{\mathbf{T}}, \hat{\mathbf{N}})$ is nasty — **Four equations** with **Three constraints** from orthonormality, but just **One** true degree of freedom.

Major Simplification occurs in (a, b) coordinates!!

$$\dot{\hat{\mathbf{T}}} = 2\begin{bmatrix} a\dot{a} - b\dot{b} \\ a\dot{b} + b\dot{a} \end{bmatrix} = 2\begin{bmatrix} a & -b \\ b & a \end{bmatrix}\begin{bmatrix} \dot{a} \\ \dot{b} \end{bmatrix}$$

Frame Evolution in (a, b):

But this formula for $\dot{\hat{\mathbf{T}}}$ is just $\kappa\hat{\mathbf{N}}$, where

$$\kappa\hat{\mathbf{N}} = \kappa\begin{bmatrix} -2ab \\ a^2 - b^2 \end{bmatrix} = \kappa\begin{bmatrix} a & -b \\ b & a \end{bmatrix}\begin{bmatrix} -b \\ a \end{bmatrix}$$

or

$$\kappa\hat{\mathbf{N}} = \kappa\begin{bmatrix} a & -b \\ b & a \end{bmatrix}\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} a \\ b \end{bmatrix}$$

2D Quaternion Frames!

Rearranging terms, *both* $\dot{\hat{\mathbf{T}}}$ and $\dot{\hat{\mathbf{N}}}$ eqns reduce to

$$\begin{bmatrix} \dot{a} \\ \dot{b} \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 & -\kappa \\ +\kappa & 0 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix}$$

This is the square root *of frame equations.*

2D Quaternions . . .

So *one equation* in the two *"quaternion"* variables (a, b) with the constraint $a^2 + b^2 = 1$ contains *both* the frame equations

$$\dot{\hat{\mathbf{T}}} = +\kappa\hat{\mathbf{N}}$$

$$\dot{\hat{\mathbf{N}}} = -\kappa\hat{\mathbf{T}}$$

$\Rightarrow$ this is much better for computer implementation, etc.

Rotation as Complex Multiplication

If we let $(a + ib) = \exp(i\,\theta/2)$ we see that rotation is complex multiplication!

"Quaternion Frames" in 2D are just complex numbers, with

Evolution Eqns = derivative of $\exp(i\,\theta/2)$!

Rotation with no matrices!

Due to an extremely deep reason in Clifford Algebras,

$$a + ib = e^{i\theta/2}$$

represents rotations "more nicely" than the matrices $R(\theta)$.

$$(a' + ib')(a + ib) = e^{i(\theta' + \theta)/2} = A + iB$$

where if we *want* the matrix, we write:

$$R(\theta')R(\theta) = R(\theta' + \theta) = \begin{bmatrix} A^2 - B^2 & -2AB \\ 2AB & A^2 - B^2 \end{bmatrix}$$

The Algebra of 2D Rotations

The algebra corresponding to 2D rotations is easy: just complex multiplication!!

$$\begin{aligned}(a',b')*(a,b) &\cong (a'+ib')(a+ib)\\ &= a'a-b'b+i(a'b+ab')\\ &\cong (a'a-b'b,\ a'b+ab')\\ &= (A,\ B)\end{aligned}$$

2D Rotations are just **complex multiplication**, and take you around the unit circle!

Quaternion Frames

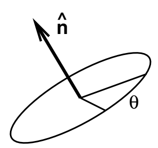
In 3D, repeat our trick: take square root of the frame, but now use *quaternions*:

- **Write down the 3D frame.**
- **Write as double-valued quadratic form.**
- **Rewrite frame evolution equations linearly in the new variables.**

The Geometry of 3D Rotations

We begin with a basic fact:

Euler theorem: *every* 3D frame can be written as a spinning by θ about a fixed axis $\hat{\mathbf{n}}$, the eigenvector of the rotation matrix:



Quaternion Frames . . .

The Matrix $R_3(\theta,\hat{\mathbf{n}})$ giving 3D rotation by θ about axis $\hat{\mathbf{n}}$ is :

$$\begin{bmatrix} c+(n_1)^2(1-c) & n_1n_2(1-c)-sn_3 & n_3n_1(1-c)+sn_2\\ n_1n_2(1-c)+sn_3 & c+(n_2)^2(1-c) & n_3n_2(1-c)-sn_1\\ n_1n_3(1-c)-sn_2 & n_2n_3(1-c)+sn_1 & c+(n_3)^2(1-c)\end{bmatrix}$$

where $c=\cos\theta$, $s=\sin\theta$, and $\hat{\mathbf{n}}\cdot\hat{\mathbf{n}}=1$.

Can we find a 720-degree form?

Remember 2D: $a^2+b^2=1$
then substitute $1-c=(a^2+b^2)-(a^2-b^2)=2b^2$
to find the remarkable expression for $\mathbf{R}(\theta,\hat{\mathbf{n}})$:

$$\begin{bmatrix} a^2-b^2+(n_1)^2(2b^2) & 2b^2n_1n_2-2abn_3 & 2b^2n_3n_1+2abn_2\\ 2b^2n_1n_2+2abn_3 & a^2-b^2+(n_2)^2(2b^2) & 2b^2n_2n_3-2abn_1\\ 2b^2n_3n_1-2abn_2 & 2b^2n_2n_3+2abn_1 & a^2-b^2+(n_3)^2(2b^2)\end{bmatrix}$$

Rotations and Quadratic Polynomials

Remember $(n_1)^2+(n_2)^2+(n_3)^2=1$ and $a^2+b^2=1$; letting

$$q_0=a=\cos(\theta/2)\qquad \mathbf{q}=b\hat{\mathbf{n}}=\hat{\mathbf{n}}\sin(\theta/2)$$

We find a matrix $R_3(q)$

$$\begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2q_1q_2-2q_0q_3 & 2q_1q_3+2q_0q_2\\ 2q_1q_2+2q_0q_3 & q_0^2-q_1^2+q_2^2-q_3^2 & 2q_2q_3-2q_0q_1\\ 2q_1q_3-2q_0q_2 & 2q_2q_3+2q_0q_1 & q_0^2-q_1^2-q_2^2+q_3^2\end{bmatrix}$$

Quaternions and Rotations . . .

HOW does $q = (q_0, \mathrm{q})$ represent rotations?

LOOK at

$$R_3(\theta, \hat{\mathrm{n}}) \stackrel{?}{=} R_3(q_0, q_1, q_2, q_3)$$

THEN we can verify that choosing

$$q(\theta, \hat{\mathrm{n}}) = (\cos\frac{\theta}{2},\ \hat{\mathrm{n}}\sin\frac{\theta}{2})$$

makes the R_3 equation an *IDENTITY*.

Quaternions and Rotations . . .

WHAT happens if you do **TWO** rotations?

EXAMINE the action of two rotations

$$R(q')R(q) = R(Q)$$

EXPRESS in quadratic forms in q and LOOK FOR an analog of complex multiplication:

Quaternions and Rotations . . .

RESULT: the following multiplication rule $\boxed{q' * q = Q}$ yields **exactly** the correct 3×3 rotation matrix $R(Q)$:

$$\begin{bmatrix} Q_0 = [q' * q]_0 \\ Q_1 = [q' * q]_1 \\ Q_2 = [q' * q]_2 \\ Q_3 = [q' * q]_3 \end{bmatrix} = \begin{bmatrix} q'_0 q_0 - q'_1 q_1 - q'_2 q_2 - q'_3 q_3 \\ q'_0 q_1 + q'_1 q_0 + q'_2 q_3 - q'_3 q_2 \\ q'_0 q_2 + q'_2 q_0 + q'_3 q_1 - q'_1 q_3 \\ q'_0 q_3 + q'_3 q_0 + q'_1 q_2 - q'_2 q_1 \end{bmatrix}$$

This is Quaternion Multiplication.

Algebra of Quaternions

= 3D Rotations!

2D rotation matrices are represented by **complex multiplication**

3D rotation matrices are represented by **quaternion multiplication**

Algebraic 2D/3D Rotations

Therefore in 3D, the 2D complex multiplication

$$(a', b') * (a, b) = (a'a - b'b,\ a'b + ab')$$

is replaced by 4D quaternion multiplication:

$$\begin{aligned} q' * q \;=\; (&q'_0 q_0 - q'_1 q_1 - q'_2 q_2 - q'_3 q_3, \\ &q'_0 q_1 + q'_1 q_0 + q'_2 q_3 - q'_3 q_2, \\ &q'_0 q_2 + q'_2 q_0 + q'_3 q_1 - q'_1 q_3, \\ &q'_0 q_3 + q'_3 q_0 + q'_1 q_2 - q'_2 q_1) \end{aligned}$$

Algebra of Quaternions . . .

The is easier to remember by dividing it into the *scalar* piece q_0 and the *vector* piece $\vec{\mathrm{q}}$:

$$\begin{aligned} q' * q \;=\; &(q'_0 q_0 - \vec{\mathrm{q}}' \cdot \vec{\mathrm{q}}, \\ &q'_0 \vec{\mathrm{q}} + q_0 \vec{\mathrm{q}}' + \vec{\mathrm{q}}' \times \vec{\mathrm{q}}) \end{aligned}$$

Now we can SEE quaternions!

Since $(q_0)^2 + \mathbf{q}\cdot\mathbf{q} = 1$ then

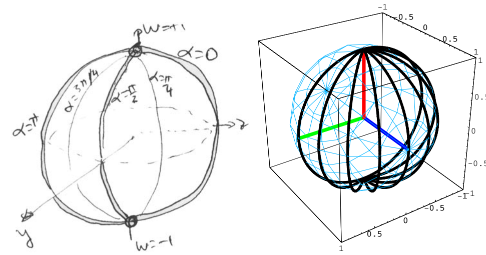
$$q_0 = \sqrt{1 - \mathbf{q}\cdot\mathbf{q}}$$

Plot just the 3D vector: $\mathbf{q} = (q_x,\, q_y,\, q_z)$

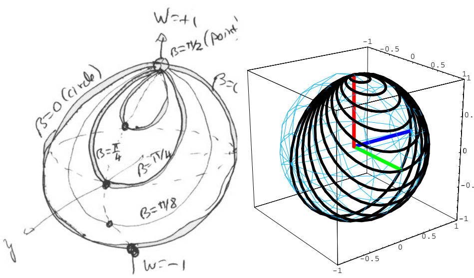
q_0 is KNOWN! Can also use any other triple: the fourth component is dependent.

DEMO

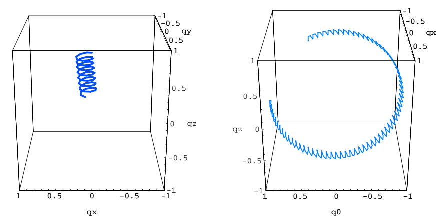
360° Belt Trick in Quaternion Form



720° Belt Trick in Quaternion Form

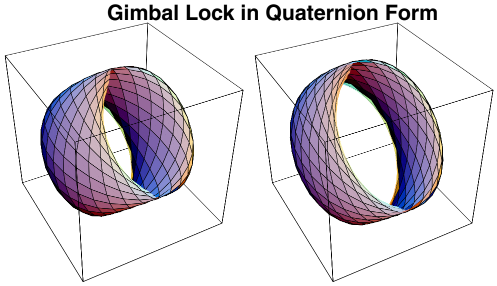


Rolling Ball in Quaternion Form



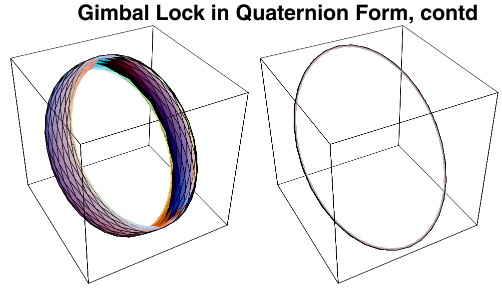
$\mathbf{q}$ vector-only plot. (q_0, q_x, q_z) plot

Gimbal Lock in Quaternion Form



Quaternion Plot of the *remaining* orientation degrees of freedom of $\mathbf{R}(\theta,\hat{\mathbf{x}})\cdot\mathbf{R}(\phi,\hat{\mathbf{y}})\cdot\mathbf{R}(\psi,\hat{\mathbf{z}})$ at $\phi = 0$ and $\phi = \pi/6$

Gimbal Lock in Quaternion Form, contd



Choosing ϕ and plotting the *remaining* orientation degrees in the rotation sequence $\mathbf{R}(\theta,\hat{\mathbf{x}})\cdot\mathbf{R}(\phi,\hat{\mathbf{y}})\cdot\mathbf{R}(\psi,\hat{\mathbf{z}})$, we see degrees of freedom **decrease from TWO to ONE as** $\phi \to \pi/2$

Quaternion Interpolations

- Shoemake (Siggraph ’85) proposed using quaternions instead of Euler angles to get smooth frame interpolations without **Gimbal Lock**:

BEST CHOICE: Animate objects and cameras using rotations represented on S^3 by quaternions

Interpolating on Spheres

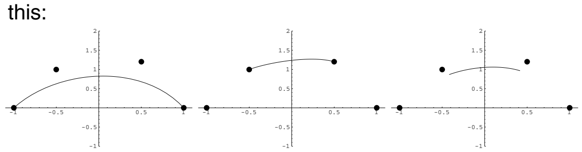
General quaternion spherical interpolation employs the “SLERP,” a constant angular velocity transition between two directions, $\hat{q}_1$ and $\hat{q}_2$:

$$\begin{aligned} \hat{q}_{12}(t) &= \text{Slerp}(\hat{q}_1, \hat{q}_2, t) \\ &= \hat{q}_1 \frac{\sin((1-t)\theta)}{\sin(\theta)} + \hat{q}_2 \frac{\sin(t\theta)}{\sin(\theta)} \end{aligned}$$

where $\cos\theta = \hat{q}_1 \cdot \hat{q}_2$.

Plane Interpolations

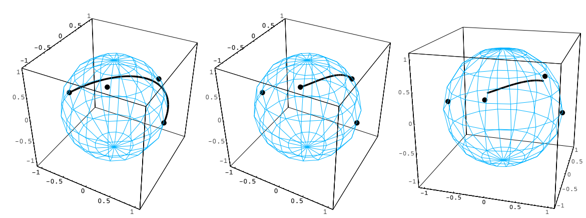
In Euclidean space, these three basic cubic splines look like this:



Bezier Catmull-Rom Uniform B

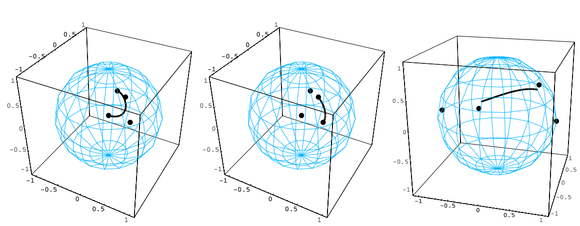
The differences are in the derivatives: Bezier has to start matching all over at every fourth point; Catmull-Rom matches the first derivative; and B-spline is the cadillac, matching all derivatives but no *control points*.

Spherical Interpolations



Bezier Catmull-Rom Uniform B

Quaternion Interpolations



Bezier Catmull-Rom Uniform B

Exp Form of Quaternion Rotations

In Hamilton’s notation, we can generalize the 2D equation

$$a + ib = e^{i\theta/2}$$

Just set

$$\begin{aligned} q &= (q_0,\ q_1,\ q_2,\ q_3) \\ &= q_0 + \mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3 \\ &= e^{(\mathbf{I}\cdot\hat{\mathbf{n}}\theta/2)} \end{aligned}$$

with $q_0 = \cos(\theta/2)$ and $\vec{q} = \hat{n}\sin(\theta/2)$ and $\mathbf{I} = (\mathbf{i}, \mathbf{j}, \mathbf{k})$, with $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = -1$, and $\mathbf{i} * \mathbf{j} = \mathbf{k}$ (cyclic),

Key to Quaternion Intuition

Fundamental Intuition: We know

$$q_0 = \cos(\theta/2), \quad \vec{q} = \hat{n}\sin(\theta/2)$$

We also know that *any coordinate frame* M can be written as $M = R(\theta, \hat{n})$.

Therefore

$\vec{q}$ points exactly along the axis we have to rotate around to go from identity I to M, and the length of $\vec{q}$ tells us how much to rotate.

Summarize Quaternion Properties

- **Unit four-vector.** Take $q = (q_0, q_1, q_2, q_3) = (q_0, \vec{q})$ to obey constraint $q \cdot q = 1$.

- **Multiplication rule.** The quaternion product q and p is $q * p = (q_0 p_0 - \vec{q}\cdot\vec{p},\ q_0\vec{p} + p_0\vec{q} + \vec{q}\times\vec{p})$, or, alternatively,

$$\begin{bmatrix} [q*p]_0 \\ [q*p]_1 \\ [q*p]_2 \\ [q*p]_3 \end{bmatrix} = \begin{bmatrix} q_0p_0 - q_1p_1 - q_2p_2 - q_3p_3 \\ q_0p_1 + q_1p_0 + q_2p_3 - q_3p_2 \\ q_0p_2 + q_2p_0 + q_3p_1 - q_1p_3 \\ q_0p_3 + q_3p_0 + q_1p_2 - q_2p_1 \end{bmatrix}$$

Quaternion Summary . . .

- **Rotation Correspondence.** The unit quaternions q and $-q$ correspond to a single 3D rotation R_3:

$$\begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2q_1q_2 - 2q_0q_3 & 2q_1q_3 + 2q_0q_2 \\ 2q_1q_2 + 2q_0q_3 & q_0^2-q_1^2+q_2^2-q_3^2 & 2q_2q_3 - 2q_0q_1 \\ 2q_1q_3 - 2q_0q_2 & 2q_2q_3 + 2q_0q_1 & q_0^2-q_1^2-q_2^2+q_3^2 \end{bmatrix}$$

If

$$q = (\cos\frac{\theta}{2}, \hat{n}\sin\frac{\theta}{2}),$$

with $\hat{n}$ a unit 3-vector, $\hat{n}\cdot\hat{n} = 1$. Then $R(\theta, \hat{n})$ is usual 3D rotation by θ in the plane $\perp$ to $\hat{n}$.

SUMMARY

- **Quaternions represent $\boxed{\text{3D frames}}$**

- **Quaternion multiplication represents 3D rotation**

- **Quaternions are]fboxpoints on a hypersphere**

- **Quaternions paths can be visualized with 3D display**

- **Belt Trick, Rolling Ball, and Gimbal Lock can be understood as Quaternion Paths.**