Visualizing Quaternions

Andrew J. Hanson

Computer Science Department

Indiana University

Siggraph 2001 Tutorial

GRAND PLAN

I: Fundamentals of Quaternions

II: Visualizing Quaternion Geometry

III: Quaternion Frames

IV: Clifford Algebras

I: Fundamentals of Quaternions

- **Motivation**
- **2D Frames:** Simple example, complex numbers.
- **3D Frames:** Rotations and quaternions.

II: Visualizing Quaternion Geometry

- **The Spherical Projection Trick:** Visualizing unit vectors.
- **Quaternion Frames**
- **Quaternion Curves**
- **Quaternion Splines**

III: Quaternion Frames

- **Quaternion Curves:** generalize the Frenet Frame
- **Quaternion Frame Evolution**
- **Quaternion Curve and Surface Optimization**

IV: Clifford Algebras

- **Clifford Algebras:** Generalize quaternion structure to N-dimensions
- **Reflections and Rotations:** New ways of looking at rotations
- **Pin(N), Spin(N), O(N), and SO(N)**

Visualizing Quaternions

Part I: Fundamentals of Quaternions

Andrew J. Hanson

Indiana University

Part I: OUTLINE

- **Motivation**
- **2D Frames:** Simple example, complex numbers.
- **3D Frames:** Rotations and quaternions.

Motivation

- Quaternion methods are now commonplace in graphics.

- Quaternions are very geometric, but we seldom attempt to visualize their properties geometrically.

- That's going to be our job today!

Basic Issues

- Basic fact number 1: Rotation matrices are **Co-ordinate Frame Axes.**

- Basic fact number 2: Rotation matrices form *groups*, which have *geometric properties*.

Task vs Strategy

- **Our task: Understand Rotations**.

- Rotations don't just **act** on geometry, rotations **are** geometry.

- **Our strategy: the geometry should help us** to visualize the properties of rotations.

Simple Example: 2D Rotations

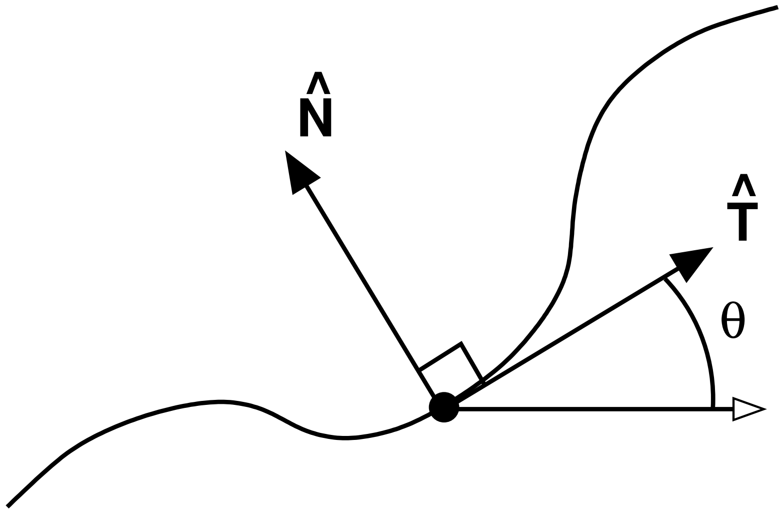
- 2D rotations $\Rightarrow$ **geometric origin** for *complex numbers.*

- **Complex numbers** are a special subspace of quaternions.

- Thus 2D rotations **introduce us to quaternions** and their geometric meaning.

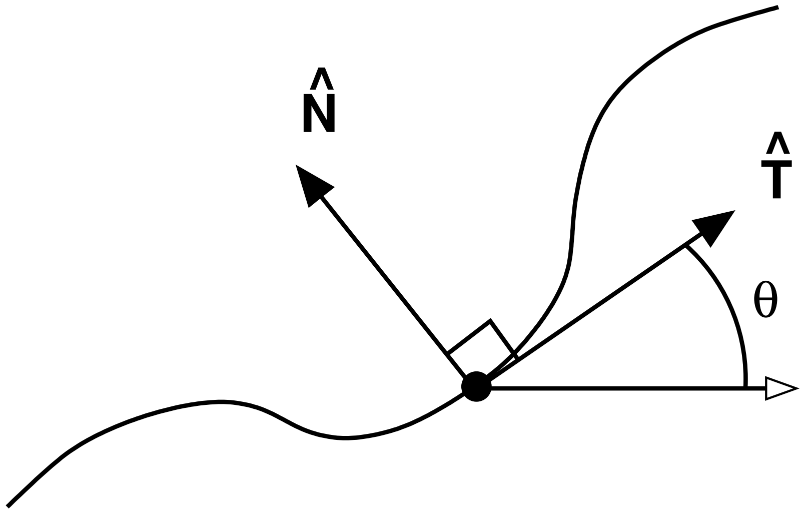
Frames in 2D

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



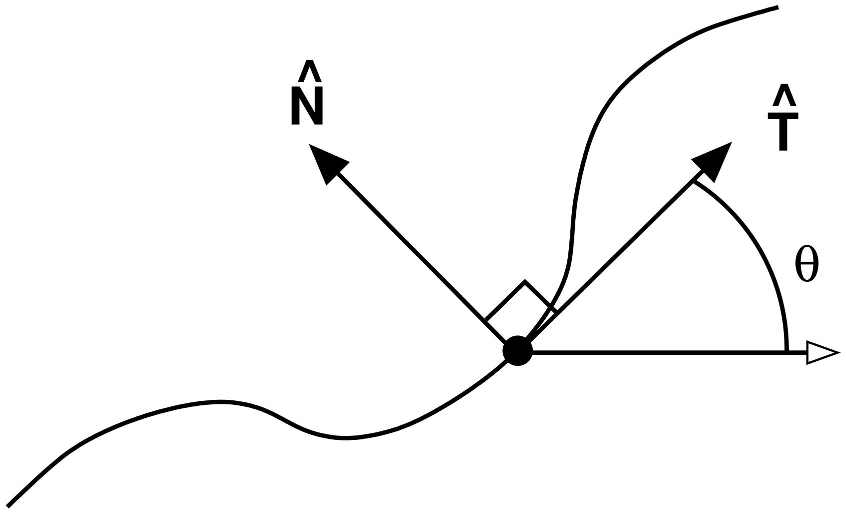
Frames in 2D

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:



Frame Matrix in 2D

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

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

Another 2D Frame

If we did not know about $\cos^2\theta + \sin^2\theta = 1$, we might represent the frame differently, e.g., as:

$$R_2(A, B) = \begin{bmatrix} A & -B \\ B & A \end{bmatrix}.$$

with the constraint $A^2 + B^2 = 1$.

The Belt Trick:

Is There a Problem?

Demonstration: Rotations "want to be doubled" to get back where you started.

See: Hart, Francis, and Kauffman.

Half-Angle Transform:

A Fix for the Problem?

$$R_2(\theta) = \begin{bmatrix} \cos^2\frac{\theta}{2} - \sin^2\frac{\theta}{2} & -2\cos\frac{\theta}{2}\sin\frac{\theta}{2} \\ \\ 2\cos\frac{\theta}{2}\sin\frac{\theta}{2} & \cos^2\frac{\theta}{2} - \sin^2\frac{\theta}{2} \\ \\ \end{bmatrix}$$

Half-Angle Transform:

A Fix for the Problem?

Or, with $a = \cos(\theta/2)$, $b = \sin(\theta/2)$,

(i.e., $A = a^2 - b^2$, $B = 2ab$),

we could parameterize 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 $a^2 + b^2 = 1$.

Half-Angle Transform:

So the pair (a, b) provides an odd **double-valued** parameterization of the frame:

$$\left[\ \hat{\mathbf{T}}\ \ \hat{\mathbf{N}}\ \right] = \begin{bmatrix} a^2 - b^2 & -2ab \\ 2ab & a^2 - b^2 \end{bmatrix}.$$

where $(a,\ b)$ is precisely the *same frame* as $(-a,\ -b)$.

Frame Evolution in 2D

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

$$\left[\ \hat{\mathbf{T}}\ \ \hat{\mathbf{N}}\ \right] = \left[\begin{array}{cc} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{array}\right] .$$

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**.

Frame Evolution in 2D

Rearrange to make a “vector matrix:”

$$\begin{bmatrix} \dot{\hat{\mathbf{T}}}(t) \\ \dot{\hat{\mathbf{N}}}(t) \end{bmatrix} = \begin{bmatrix} 0 & +\kappa(t) \\ -\kappa(t) & 0 \end{bmatrix} \begin{bmatrix} \hat{\mathbf{T}}(t) \\ \hat{\mathbf{N}}(t) \end{bmatrix}$$

Frame Evolution in (a, b):

Using the basis $(\hat{\mathbf{T}}, \hat{\mathbf{N}})$ we have **Four equations** with **Three constraints** from orthonormality, for **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 $\kappa\widehat{\mathbf{N}}$, where

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

or

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

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

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

Rotation with no matrices!

This is the miracle:

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

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:

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

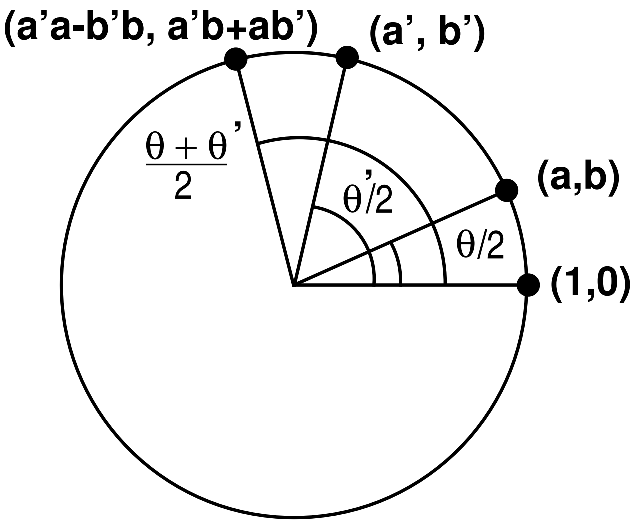
The Algebra of 2D Rotations

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

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

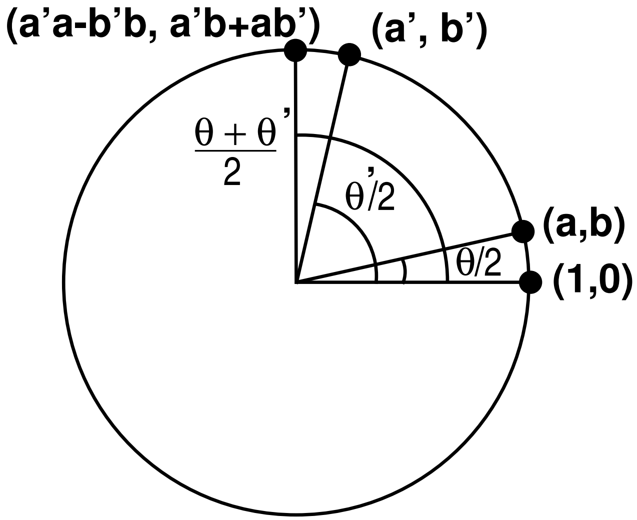
The Geometry of 2D Rotations

(a, b) with $a^2 + b^2 = 1$ is a **point on the unit circle**, also written S^1. Rotations are just **complex multiplication**, and take you around the unit circle like this:



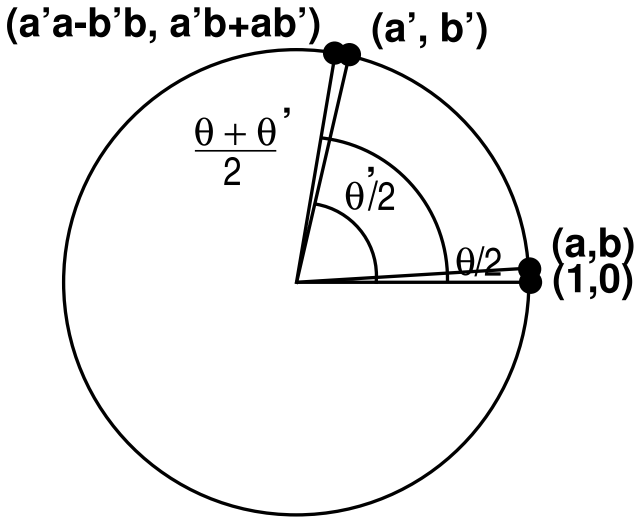
The Geometry of 2D Rotations

(a, b) with $a^2 + b^2 = 1$ is a **point on the unit circle**, also written S^1. Rotations are just **complex multiplication**, and take you around the unit circle like this:



The Geometry of 2D Rotations

(a, b) with $a^2 + b^2 = 1$ is a **point on the unit circle**, also written S^1. Rotations are just **complex multiplication**, and take you around the unit circle like this:



Quaternion Frames

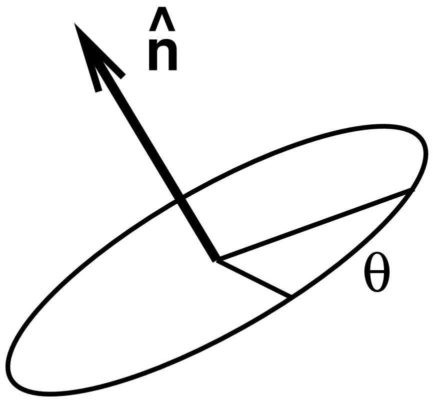
In 3D, repeat our trick: take square root of the frame, but now use *quaternions*:

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

The Geometry of 3D Rotations

We begin with a basic fact:

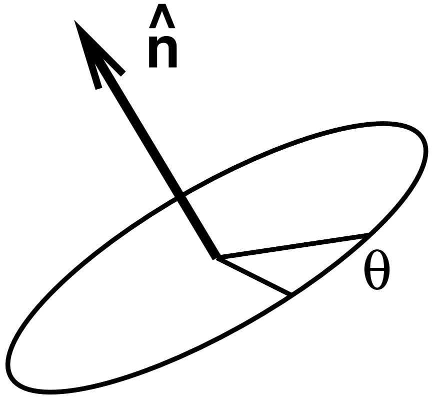
Euler theorem: *every* 3D frame can be written as a spinning by θ about a fixed axis $\hat{\mathbf{n}}$, the eigenvector of the rotation matrix:



The Geometry of 3D Rotations

We begin with a basic fact:

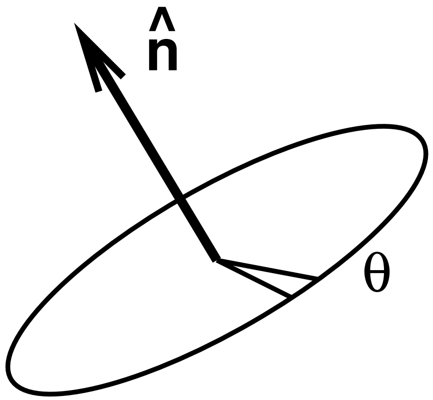
Euler theorem: *every* 3D frame can be written as a spinning by θ about a fixed axis $\widehat{\mathbf{n}}$, the eigenvector of the rotation matrix:



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 . . .

Matrix giving 3D rotation by θ about axis $\hat{\mathbf{n}}$:

$$R_3(\theta, \hat{\mathbf{n}}) =$$

$$\begin{bmatrix} c + (n_1)^2(1-c) & n_1 n_2(1-c) - s n_3 & n_3 n_1(1-c) + s n_2 \\ n_1 n_2(1-c) + s n_3 & c + (n_2)^2(1-c) & n_3 n_2(1-c) - s n_1 \\ n_1 n_3(1-c) - s n_2 & n_2 n_3(1-c) + s n_1 & c + (n_3)^2(1-c) \end{bmatrix}$$

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

Quaternion Frame Parameters

To find θ and axis $\hat{\mathbf{n}}$, given *any* rotation matrix or frame M, we need two steps:

$$\mathrm{Tr}M = 1 + 2\cos\theta$$

$\Rightarrow$ solve for θ.

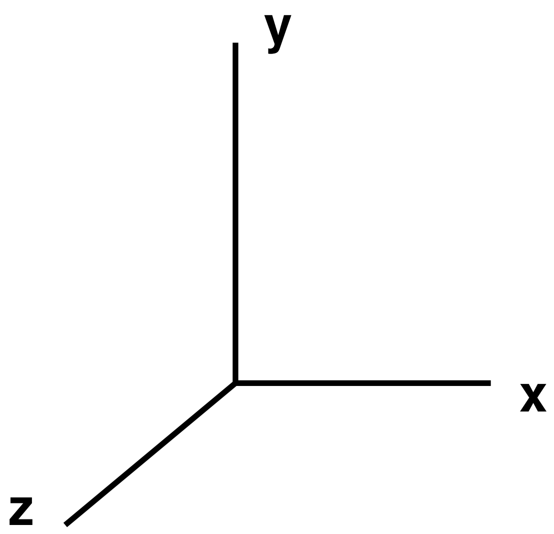
$$M - M^t =$$

$$\begin{bmatrix} 0 & -2n_3\sin\theta & +2n_2\sin\theta \\ +2n_3\sin\theta & 0 & -2n_1\sin\theta \\ -2n_2\sin\theta & +2n_1\sin\theta & 0 \end{bmatrix}$$

$\Rightarrow$ solve for $\hat{\mathbf{n}}$ as long as $\theta \neq 0$.

Quaternions and Rotations

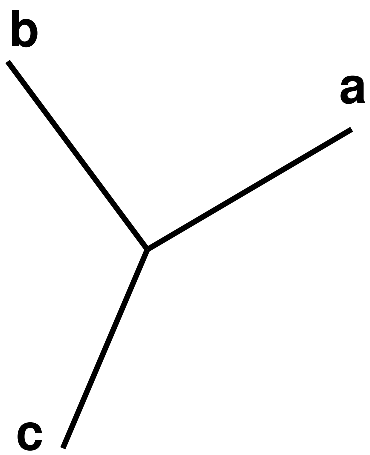
Some set of axes can be chosen as the identity matrix:



$$= \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Quaternions and Rotations

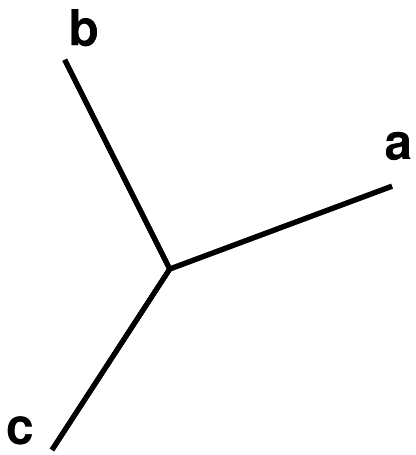
Any arbitrary set of axes forms the columns of an orthogonal rotation matrix:



$$= \begin{vmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{vmatrix}$$

Quaternions and Rotations

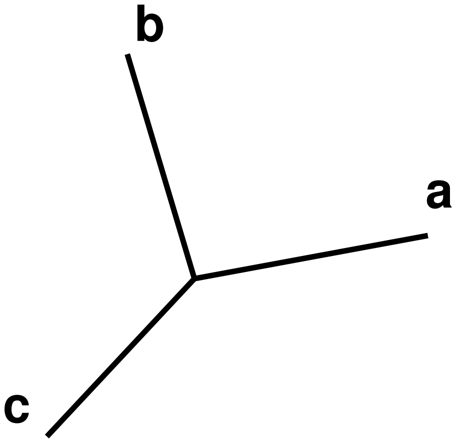
Any arbitrary set of axes forms the columns of an orthogonal rotation matrix:



$$= \begin{vmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{vmatrix}$$

Quaternions and Rotations

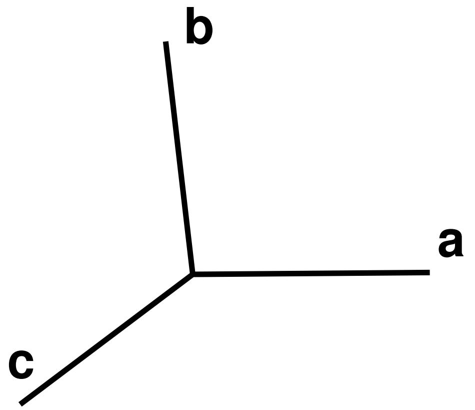
Any arbitrary set of axes forms the columns of an orthogonal rotation matrix:



$$= \begin{vmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{vmatrix}$$

Quaternions and Rotations

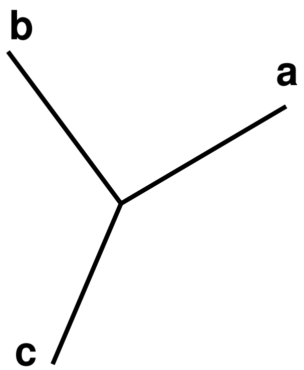
Any arbitrary set of axes forms the columns of an orthogonal rotation matrix:



$$= \begin{vmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{vmatrix}$$

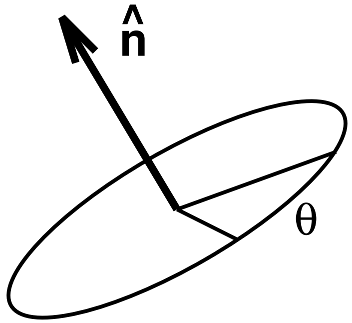
Quaternions and Rotations

By Euler's theorem, that matrix has an *eigenvector* $\hat{\mathbf{n}}$, and so is representable as a single rotation about $\hat{\mathbf{n}}$ applied to the identity:

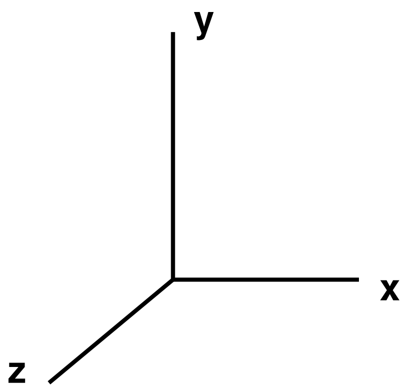


$$= \begin{vmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{vmatrix}$$

$=$



$\times$



$$= \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Rotations and Quadratic Polynomials

Remember $R_2(\theta) = \begin{bmatrix} a^2 - b^2 & -2ab \\ 2ab & a^2 - b^2 \end{bmatrix}$?

What if we try a 3×3 matrix R_3 instead of 2×2?

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

Hint: set $q_1 = q_2 = 0$ or any other $(i \neq j)$ pair to see a familiar sight!

Quaternions and Rotations

Why does this matrix parameterize a rotation? Because *Columns* of $R_3(q_0, q_1, q_2, q_3)$ are orthogonal:

$$\mathrm{col}_i \cdot \mathrm{col}_j = 0 \text{ for } i \neq j$$

What is LENGTH of 3-vector column?

$$\mathrm{col}_i \cdot \mathrm{col}_i = (q_0^2 + q_1^2 + q_2^2 + q_3^2)^2$$

Quaternions and Rotations . . .

So if we require $\boxed{q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1}$, orthonormality is assured and $R_3(q_0, q_1, q_2, q_3)$ is a rotation.

This implies q **is a point on 3-sphere in 4D**.

NOTE: $q \Rightarrow -q$ gives *same* $R_3()$.

Quaternions and Rotations . . .

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

LOOK at

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

NOTICE: Choosing

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

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:

RESULT: the following multiplication rule $q' * q = Q$ 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 $\vec{\mathbf{q}}$:

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

Quaternions and Rotations

Another miracle: let us generalize the 2D equation

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

How? We 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 $\vec{\mathbf{q}} = \hat{\mathbf{n}}\sin(\theta/2)$ and $\mathbf{I} = (\mathbf{i}, \mathbf{j}, \mathbf{k})$.

Quaternions and Rotations ...

Then if we take $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = -1$, and $\mathbf{i} * \mathbf{j} = \mathbf{k}$ (cyclic), quaternion multiplication rule is automatic!

$\Rightarrow \; q = q_0 + \mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3$ is the standard representation for a *quaternion*, and we can also use 2×2 Pauli matrices in place of $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ if we want.

Key to Quaternion Intuition

Fundamental Intuition: We know

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

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

Therefore

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

Summarize Quaternion Properties

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

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

$$\begin{bmatrix} [q*p]_0 \\ [q*p]_1 \\ [q*p]_2 \\ [q*p]_3 \end{bmatrix} = \begin{bmatrix} q_0 p_0 - q_1 p_1 - q_2 p_2 - q_3 p_3 \\ q_0 p_1 + q_1 p_0 + q_2 p_3 - q_3 p_2 \\ q_0 p_2 + q_2 p_0 + q_3 p_1 - q_1 p_3 \\ q_0 p_3 + q_3 p_0 + q_1 p_2 - q_2 p_1 \end{bmatrix}$$

Quaternion Summary . . .

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

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

- **Rotation Correspondence.** Let

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

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

- **Inversion.** Any 3×3 matrix R can be inverted for q *up to a sign*. Carefully treat singularities! Can choose sign, e.g., by local consistency, to get continuous frames.

SUMMARY

- **Complex numbers** represent 2D frames.
- **Complex multiplication represents 2D rotation.**
- **Quaternions** represent 3D frames.
- **Quaternion multiplication represents 3D rotation.**