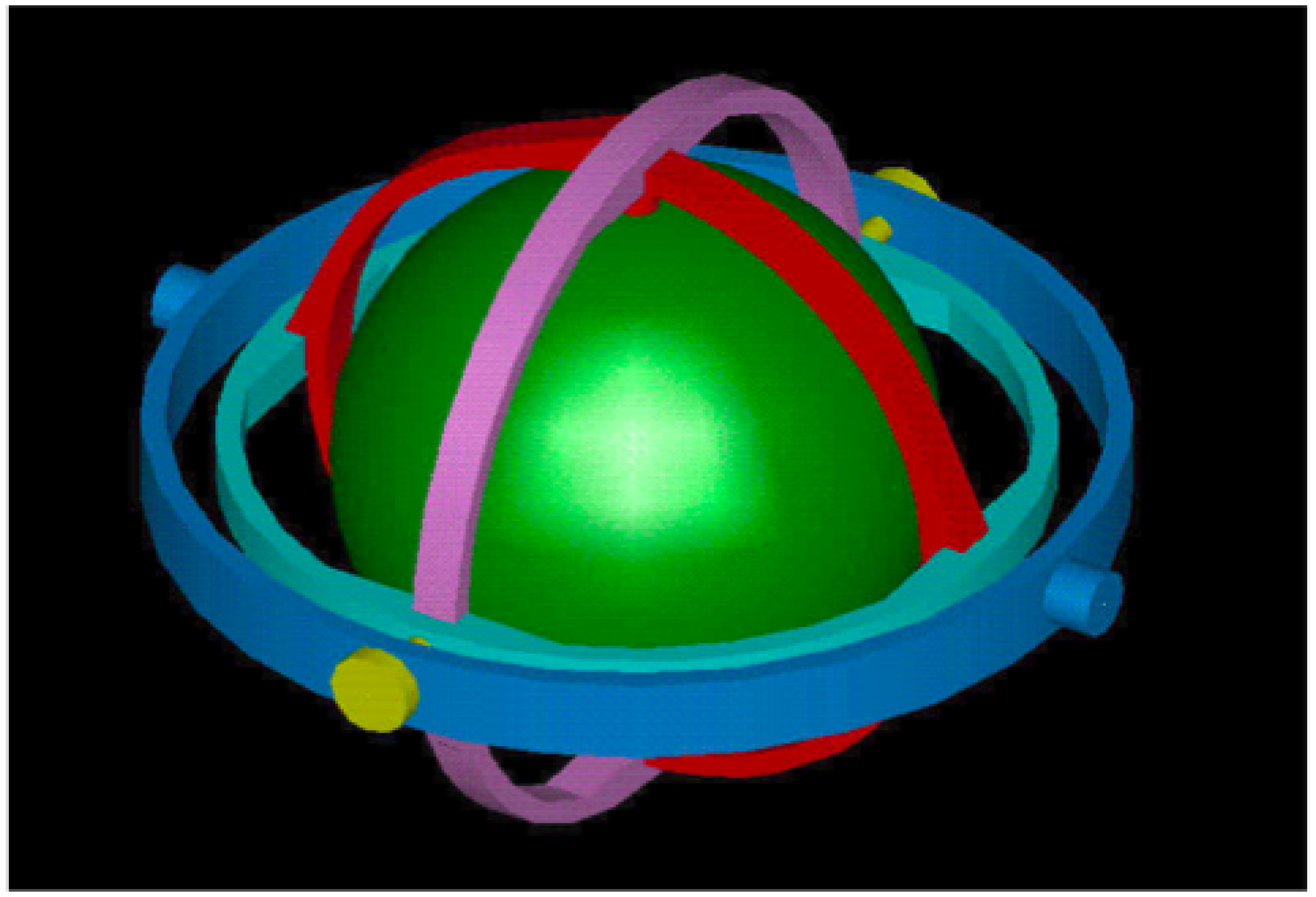
11 3D rotations and quaternions

Steve Marschner

CS5625 Spring 2022

Parameterizing rotations



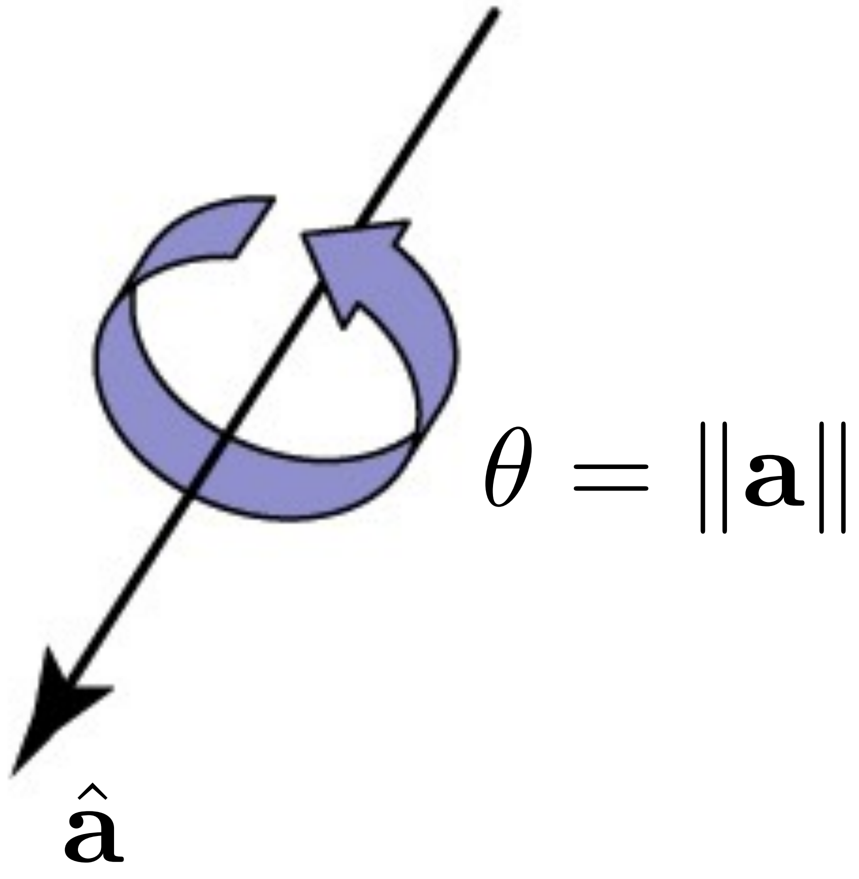
- Euler angles
  - rotate around x, then y, then z
  - nice and simple

  $$R(\theta_x, \theta_y, \theta_z) = R_z(\theta_z) R_y(\theta_y) R_x(\theta_x)$$

- Axis/angle
  - specify axis to rotate around, then angle by which to rotate

  $$R(\hat{\mathbf{a}}, \theta) = F_{\hat{\mathbf{a}}} R_x(\theta) F_{\hat{\mathbf{a}}}^{-1}$$

$F_{\hat{\mathbf{a}}}$ is a frame matrix with $\mathbf{a}$ as its first column.



- Unit quaternions
  - A 4D representation (like 3D unit vectors for 2D sphere)
  - Good choice for interpolating rotations

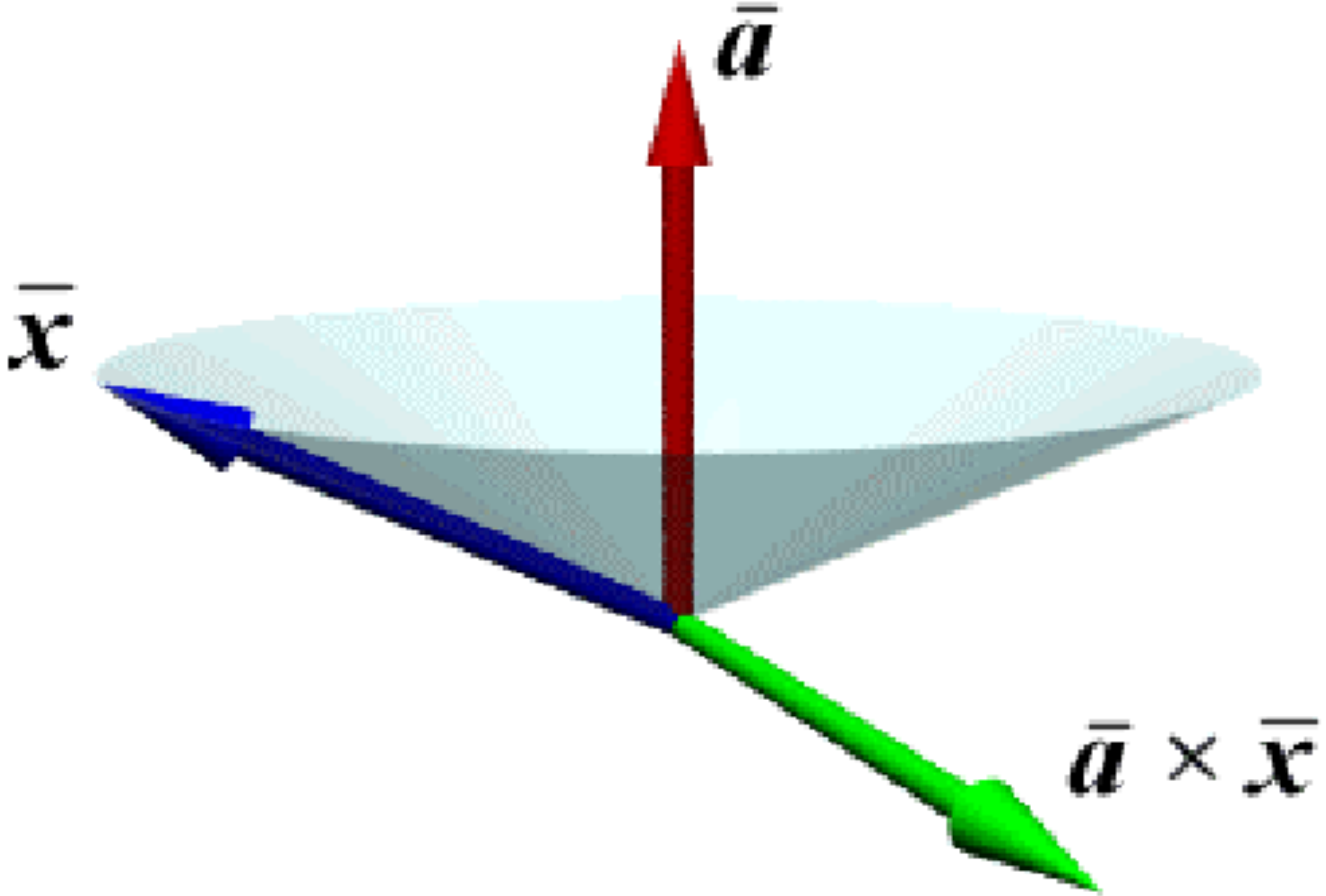
Problems

- Euler angles
  - gimbal lock (saw this before)
  - some rotations have many representations
- Axis/angle
  - multiple representations for identity rotation
  - even with combined rotation angle, making small changes near 180 degree rotations requires larger changes to parameters
- These resemble the problems with polar coordinates on the sphere
  - as with choosing poles, choosing the reference orientation for an object changes how the representation works

Rodrigues' rotation formula

$$R(\mathbf{a},\theta)\mathbf{x} = (\cos\theta)\mathbf{x} + (\sin\theta)(\mathbf{a}\times\mathbf{x}) + (1-\cos\theta)(\mathbf{a}\cdot\mathbf{x})\mathbf{a}$$

$$R(\mathbf{a},\theta) = (\cos\theta)I + (\sin\theta)\tilde{\mathbf{a}} + (1-\cos\theta)\mathbf{a}\mathbf{a}^T$$



[Leonard McMillan]

What is a rotation?

- Think of the set of possible orientations of a 3D object
  - you get from one orientation to another by rotating
  - if we agree on some starting orientation, rotations and orientations are pretty much the same thing
- It is a smoothly connected three-dimensional space
  - how can you tell? For any orientation, I can make a small rotation around any axis (pick axis = 2D, pick angle = 1D)
- This set is a subset of linear transformations called SO(3)
  - **O** for orthogonal, **S** for "special" (determinant +1), **3** for 3D

Calculating with rotations

- Representing rotations with numbers requires a function

$$f : \mathbb{R}^n \rightarrow SO(3)$$

- The situation is analogous to representing directions in 3-space
  - there we are dealing with the set S^2, the two-dimensional sphere (I mean the sphere is a 2D surface)
  - like SO(3) it is very symmetric; no directions are specially distinguished

Analogy: spherical coordinates

- We can use latitude and longitude to parameterize the 2-sphere (aka. directions in 3D), but with some annoyances
  - the poles are special, and are represented many times
  - if you are at the pole, going East does nothing
  - near the pole you have to change longitude a lot to get anywhere
  - traveling along straight lines in (latitude, longitude) leads to some pretty weird paths on the globe

    you are standing one mile from the pole, facing towards it; to get to the point 2 miles ahead of you the map tells you to turn right and walk 3.14 miles along a latitude line…
  - Conclusion: use unit vectors instead

Analogy: unit vectors

- When we want to represent directions we use unit vectors: points that are literally on the unit sphere in R^3
  - now no points are special
  - every point has a unique representation
  - equal sized changes in coordinates are equal sized changes in direction
- Down side: one too many coordinates
  - have to maintain normalization
  - but normalize() is a simple and easy operation

Complex numbers to quaternions

- Rather than one imaginary unit *i*, there are three such symbols *i*, *j*, and *k*, with the properties

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

- Multiplication of these units acts like the cross product

$$\begin{aligned} ij &= k & ji &= -k \\ jk &= i & kj &= -i \\ ki &= j & ik &= -j \end{aligned}$$

- Combining multiples of *i*, *j*, *k* with a scalar gives the general form of a quaternion:

$$\mathbf{H} = \{a + bi + cj + dk \mid (a, b, c, d) \in \mathbb{R}^4\}$$

Complex numbers to quaternions

- Like complex numbers, quaternions have conjugates and magnitudes

$$q = a + bi + cj + dk$$
$$\bar{q} = a - bi - cj - dk$$
$$|q| = (q\bar{q})^{\frac{1}{2}} = \sqrt{a^2 + b^2 + c^2 + d^2} = \|(a, b, c, d)\|$$

- Also like complex numbers, quaternions have reciprocals of the form

$$q^{-1} = \frac{1}{q} = \frac{\bar{q}}{|q|}$$

Quaternion Properties

- Associative

$$q_1(q_2q_3) = q_1q_2q_3 = (q_1q_2)q_3$$

- Not commutative

$$q_1q_2 \neq q_2q_1$$

- Magnitudes multiply

$$|q_1q_2| = |q_1|\,|q_2|$$

Unit quaternions

- The set of unit-magnitude quaternions is called the "unit quaternions"

$$S^3 = \{q \in \mathbf{H} \mid |q| = 1\}$$

  - as a subset of 4D space, it is the unit 3-sphere
  - multiplying unit quaternions produces more unit quaternions

$$|q_1| = |q_2| = 1 \implies |q_1 q_2| = 1$$

$$q_1, q_2 \in S^3 \implies q_1 q_2 \in S^3$$

- For unit quaternions:

$$|q| = 1$$

$$q^{-1} = \bar{q}$$

Quaternion as scalar plus vector

- Write q as a pair of a scalar $s \in \mathbf{R}$ and vector $\mathbf{v} \in \mathbf{R}^3$

$$q = a + bi + cj + dk$$

$$q = s + \mathbf{v} \text{ where } s = a \text{ and } \mathbf{v} = bi + cj + dk$$

$$q = (s, \mathbf{v}) \text{ where } s = a \text{ and } \mathbf{v} = (b, c, d)$$

- Multiplication: $\mathbf{v}_1\mathbf{v}_2 = -\mathbf{v}_1 \cdot \mathbf{v}_2 + \mathbf{v}_1 \times \mathbf{v}_2$

$$(s_1 + \mathbf{v}_1)(s_2 + \mathbf{v}_2) = s_1 s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2 + s_1\mathbf{v}_2 + s_2\mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2$$

$$(s_1, \mathbf{v}_1)(s_2, \mathbf{v}_2) = (s_1 s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2, s_1\mathbf{v}_2 + s_2\mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2)$$

- For a unit quaternion, $|s|^2 + \|\mathbf{v}\|^2 = 1$
  - so think of these as the sine and cosine of an angle ψ:

$$q = (\cos\psi, \hat{\mathbf{v}}\sin\psi) \text{ or } \cos\psi + \hat{\mathbf{v}}\sin\psi$$

  - this is a lot like writing a 2D rotation as $\cos\theta + i\sin\theta$

Quaternions and rotations

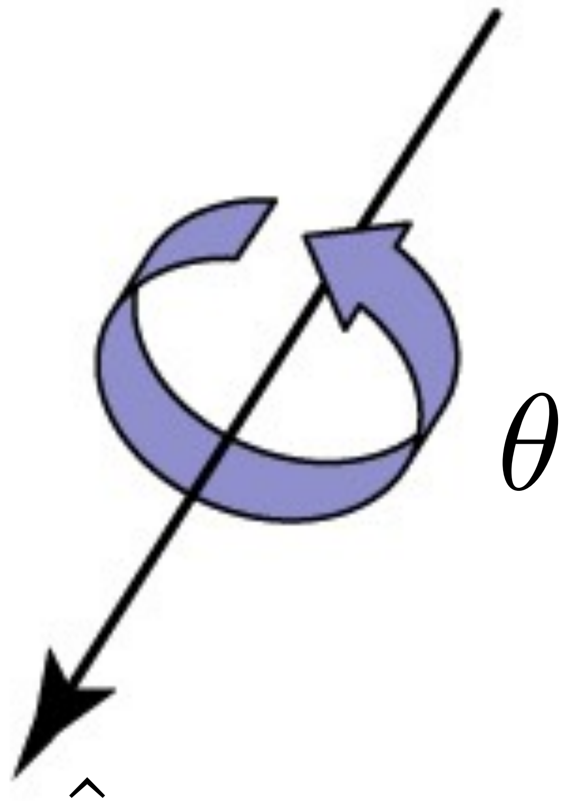
There is a natural association between the unit quaternion

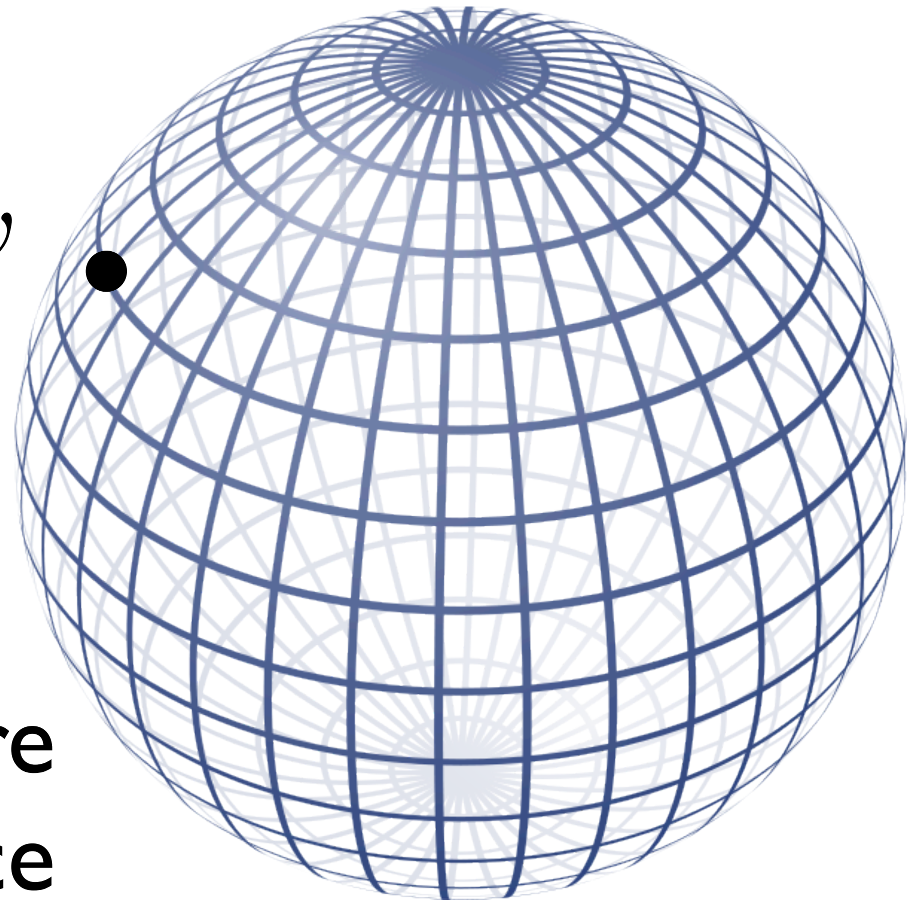
$$\cos\psi + \hat{\mathbf{v}}\sin\psi \in S^3 \subset \mathbf{H}$$

and the 3D axis-angle rotation

$$R_{\hat{\mathbf{v}}}(\theta) \in SO(3)$$

where $\theta = 2\psi$.





Note $s + \mathbf{v}$ and $-s - \mathbf{v}$ represent the same rotation

Rotation and quaternion multiplication

- Represent a point in space by a pure-imaginary quaternion

$$\mathbf{x} = (x, y, z) \in I\!R^3 \leftrightarrow X = xi + yj + zk \in \mathbf{H}$$

- Can compute rotations using quaternion multiplication

$$X_{\text{rotated}} = qX\bar{q}$$

  - note that q and $-q$ correspond to the same rotation
  - you can verify this is a rotation by multiplying out…
- Multiplication of quaternions corresponds to composition of rotations

$$q_1(q_2 X \bar{q_2})\bar{q_1} = (q_1 q_2) X (\bar{q_2}\bar{q_1}) = q_1 q_2 \, X \, \overline{q_1 q_2}$$

  - the quaternion $q_1 q_2$ corresponds to "rotate by q_2, then rotate by q_1"

Analogy: rays vs. lines

- The set of directions (unit vectors) describes the set of rays leaving a point
- The set of lines through a point is a bit different
  - no notion of "forward" vs. "backward"
- Would probably still represent using unit vectors
  - but every line has exactly two representations, **v** and –**v**
- Similarly every rotation has exactly two representations
  - $q = \cos\psi + \mathbf{v}\sin\psi;\ -q = \cos(\pi - \psi) - \mathbf{v}\sin(\pi - \psi)$
  - a rotation by the opposite angle $(2\pi - \theta)$ around the negated axis

Rotation and quaternion multiplication

If we write a unit quaternion in the form

$$q = \cos\psi + \hat{\mathbf{v}}\sin\psi$$

then the operation

$$X_{\text{rotated}} = qX\bar{q} = (\cos\psi + \hat{\mathbf{v}}\sin\psi)X(\cos\psi - \hat{\mathbf{v}}\sin\psi)$$

is a rotation by 2ψ around the axis **v**.

So an alternative explanation is, "All this algebraic mumbo-jumbo aside, a quaternion is just a slightly different way to encode an axis and an angle in four numbers: rather than the number θ and the unit vector **v**, we store the number cos (θ/2) and the vector sin (θ/2) **v**."

Unit quaternions and axis/angle

- We can write down a parameterization of 3D rotations using unit quaternions (points on the 3-sphere)

$$f : S^3 \subset \mathbf{H} \to SO(3)$$

$$: \cos\psi + \hat{\mathbf{v}}\sin\psi \mapsto R_{\hat{\mathbf{v}}}(2\psi)$$

$$: (w, x, y, z) \mapsto \begin{bmatrix} w^2 + x^2 - y^2 - z^2 & 2(xy - wz) & 2(xz + wy) \\ 2(xy + wz) & w^2 - x^2 + y^2 - z^2 & 2(yz - wx) \\ 2(xz - wy) & 2(yz + wx) & w^2 - x^2 - y^2 + z^2 \end{bmatrix}$$

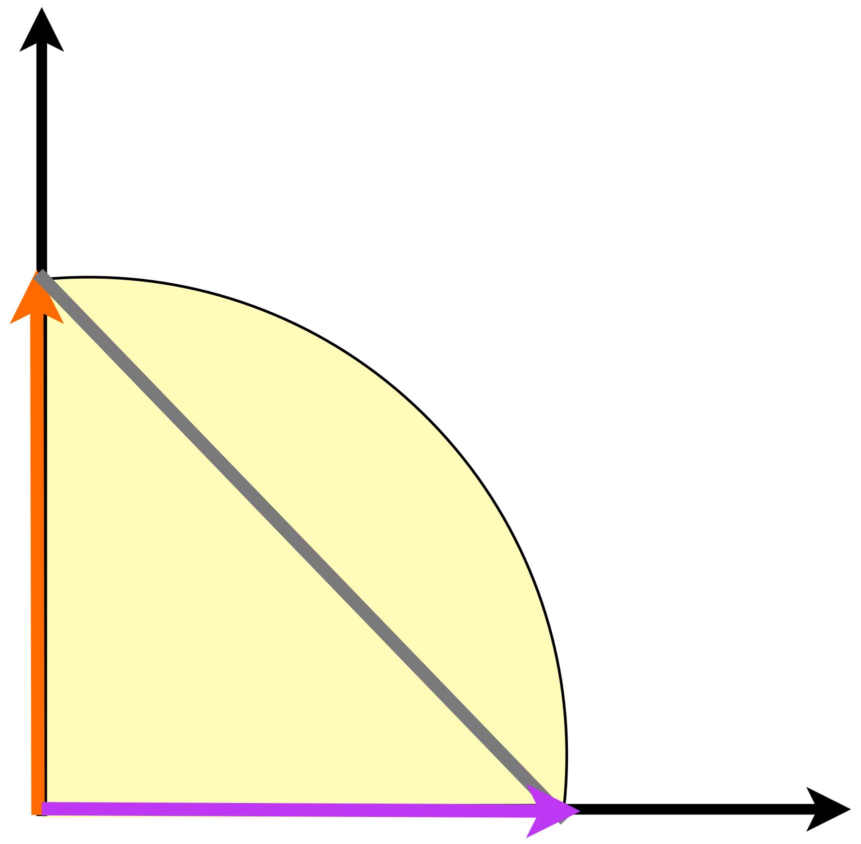
- This mapping is wonderfully uniform:
  - is exactly 2-to-1 everywhere
  - has constant speed in all directions
  - has constant Jacobian (does not distort "volume")
  - maps shortest paths to shortest paths
  - *and*… it comes with a multiplication operation (not mentioned today)

Why Quaternions?

- Fast, few operations, not redundant
- Numerically stable for incremental changes
- Composes rotations nicely
- Convert to matrices at the end
- Biggest reason: spherical interpolation

Interpolating between quaternions

- Why not linear interpolation?
  - Need to be normalized
  - Does not have constant rate of rotation



$$\frac{(1-\alpha)x + \alpha y}{||(1-\alpha)x + \alpha y||}$$

Analogy: interpolating directions

- Interpolating in the space of 3D vectors is well behaved
- Simple computation: interpolate linearly and normalize
  - this is what we do all the time, e.g. with normals for fragment shading

$$\hat{\mathbf{v}}(t) = \mathrm{normalize}((1-t)\mathbf{v}_0 + t\mathbf{v}_1)$$

  - but for far-apart endpoints the speed is uneven (faster towards the middle)
- For constant speed: spherical linear interpolation
  - build basis $\{\mathbf{v}_0, \mathbf{w}\}$ from $\mathbf{v}_0$ and $\mathbf{v}_1$
  - interpolate angle from 0 to θ
  - (slicker way in a few slides)

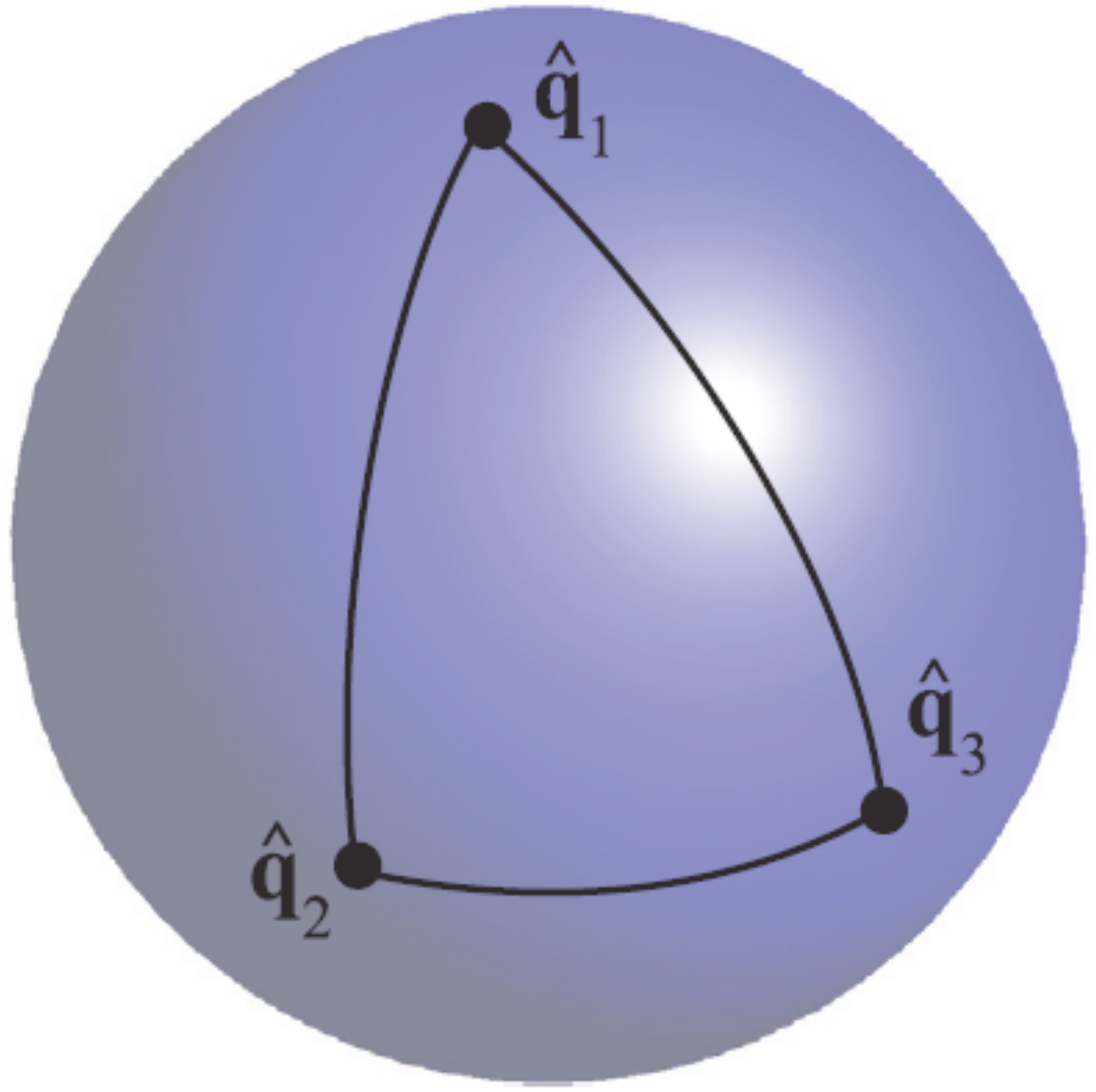
$$\begin{aligned}
\mathbf{w} &= \hat{\mathbf{v}}_1 - (\hat{\mathbf{v}}_0 \cdot \hat{\mathbf{v}}_1)\hat{\mathbf{v}}_0 \\
\hat{\mathbf{w}} &= \mathbf{w}/\|\mathbf{w}\| \\
\theta &= \mathrm{acos}(\hat{\mathbf{v}}_0 \cdot \hat{\mathbf{v}}_1) \\
\hat{\mathbf{v}}(t) &= (\cos t\theta)\, \hat{\mathbf{v}}_0 + (\sin t\theta)\, \hat{\mathbf{w}}
\end{aligned}$$

Spherical Linear Interpolation

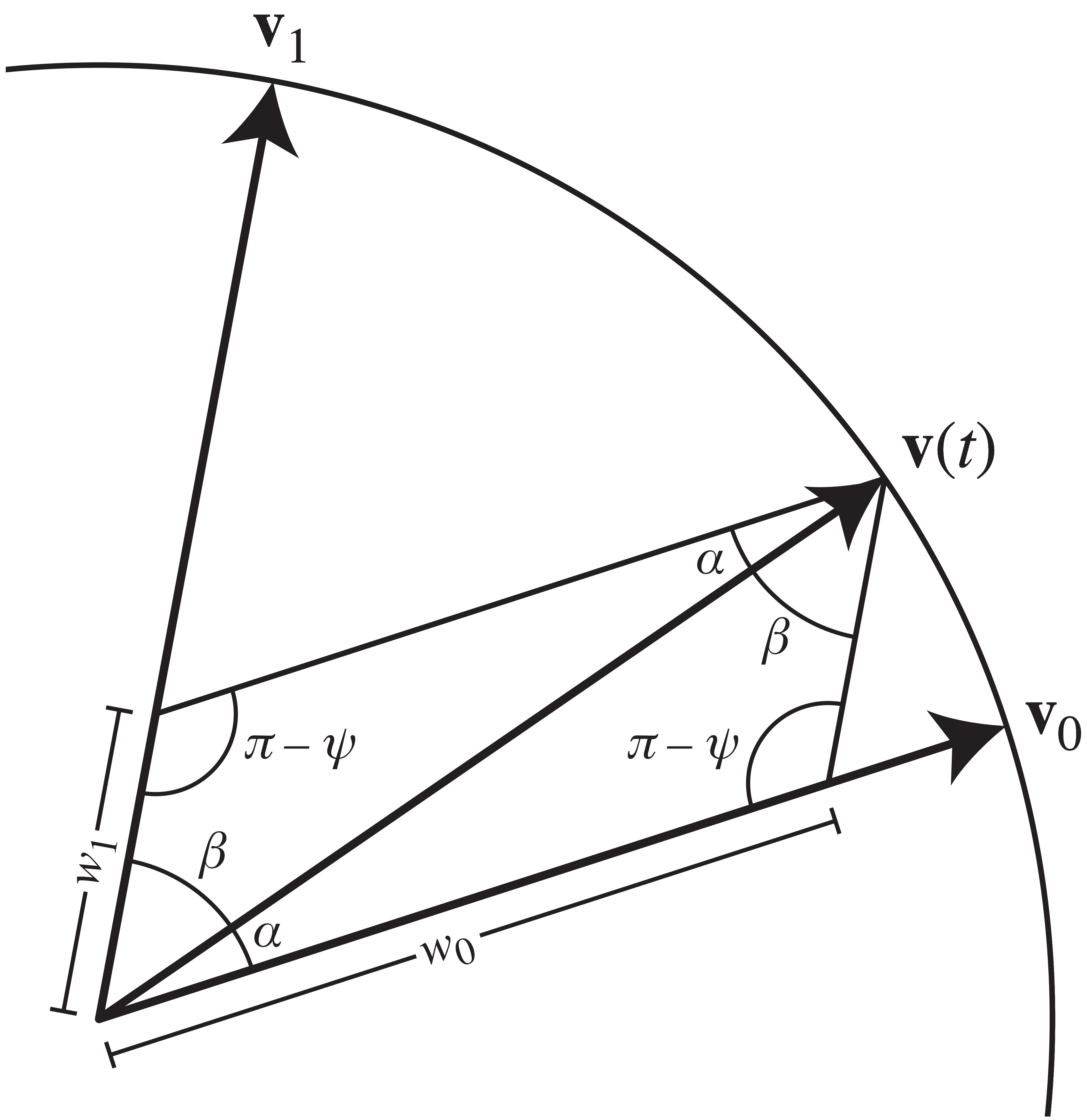
- Intuitive interpolation between different orientations
  - Nicely represented through quaternions
  - Useful for animation
  - Given two quaternions, interpolate between them

  - Shortest path between two points on sphere

    Geodesic, on Great Circle



Spherical linear interpolation ("slerp")



- given vectors $\mathbf{v}_0$, $\mathbf{v}_1$, and parameter t

$$\psi = \cos^{-1}(\mathbf{v}_0 \cdot \mathbf{v}_1)$$

$$\alpha = t\psi; \;\; \beta = (1-t)\psi$$

- express answer as a weighted sum

$$\mathbf{v}(t) = w_0\mathbf{v}_0 + w_1\mathbf{v}_1$$

- then from law of sines

$$\frac{\sin\alpha}{w_1} = \frac{\sin\beta}{w_0} = \frac{\sin(\pi-\psi)}{1} = \sin\psi$$

$$w_0 = \sin\beta / \sin\psi$$

$$w_1 = \sin\alpha / \sin\psi$$

$$\mathbf{v}(t) = \frac{\mathbf{v}_0 \sin\beta + \mathbf{v}_1 \sin\alpha}{\sin\psi}$$

Quaternion Interpolation

- Spherical linear interpolation naturally works in any dimension
- Traverses a great arc on the sphere of unit quaternions
  - Uniform angular rotation velocity about a fixed axis

$$\psi = \cos^{-1}(q_0 \cdot q_1)$$

$$q(t) = \frac{q_0 \sin(1-t)\psi + q_1 \sin t\psi}{\sin \psi}$$

- When angle gets close to zero, estimation of ψ is inaccurate
  - switch to linear interpolation when $q_0 \approx q_1$.
- q is same rotation as $-q$
  - if $q_0 \cdot q_1 > 0$, slerp between them
  - else, slerp between q_0 and $-q_1$

Dual quaternions

- One jump farther down the rabbit hole: combine quaternions with *dual numbers*
- Result is an algebraic system in which elements have 8 degrees of freedom (like quaternions have 4)
- Unit-norm dual quaternions have two constraints, so they constitute a 6D subspace
- Just as rotations can be identified with quaternions, both rotations and translations can be identified with dual quaternions
- Both linear quaternion interpolation and Slerp have analogues for dual quaternions, with similar properties

Dual numbers

- Real numbers plus a dual unit "ε" with the multiplication rule that $\varepsilon^2 = 0$

$$(a + \epsilon b)(c + \epsilon d) = ab + \epsilon(bd + ad)$$

- There is a conjugation for dual numbers

$$(a + \epsilon b)^* = a - \epsilon b$$

$$\hat{a}^* \hat{b}^* = (\hat{a}\hat{b})^*$$

caution:
my notation for the two conjugations is swapped from Kavan et al.

- Some quirky algebraic features (ex: verify!)

$$\frac{1}{a + \epsilon b} = \frac{1}{a} - \epsilon \frac{b}{a^2} \qquad \sqrt{a + \epsilon b} = \sqrt{a} + \epsilon \frac{b}{2\sqrt{a}}$$

Dual quaternions

- A d.q. is a quaternion built from dual numbers

$$\hat{q} = \hat{w} + i\hat{x} + j\hat{y} + k\hat{z}$$

- A d.q. is also a dual number built from quaternions

$$\hat{q} = q + \epsilon q'$$

- Either way there are 8 components

$$\hat{q} = w + ix + jy + kz + \epsilon w' + i\epsilon x' + j\epsilon y' + k\epsilon z'$$

  – and three ways to conjugate

$$\hat{q}^* = w + ix + jy + kz - \epsilon w' - i\epsilon x' - j\epsilon y' - k\epsilon z'$$

$$\overline{\hat{q}} = w - ix - jy - kz + \epsilon w' - i\epsilon x' - j\epsilon y' - k\epsilon z'$$

$$\overline{\hat{q}^*} = w - ix - jy - kz - \epsilon w' + i\epsilon x' + j\epsilon y' + k\epsilon z'$$

Unit dual quaternions

- Norm of a d.q. is defined as

$$\|\hat{q}\|^2 = \hat{q}\bar{\hat{q}} = (q + \epsilon p)(\bar{q} + \epsilon \bar{p}) = q\bar{q} + \epsilon(p\bar{q} + q\bar{p})$$
$$= \|q\|^2 + 2\epsilon(q \cdot p)$$

- This is a dual number, so being a unit dual quaternion requires satisfying two separate constraints:

$$\|q\| = 1$$
$$q \cdot p = 0$$

Dual quaternion transformation

- For transformation by dual quaternions, interpret the vector **v** as a pure imaginary quaternion and represent it as a dual quaternion of the form

  $$\hat{\mathbf{v}} = 1 + \epsilon \mathbf{v}$$

  and apply a transformation as with quaternions but using double conjugation:

  $$\hat{\mathbf{v}} \mapsto \hat{q}\hat{\mathbf{v}}\overline{\hat{q}^*}$$

- Transformation by an ordinary quaternion goes through:

  $$\hat{\mathbf{v}} \mapsto q(1 + \epsilon \mathbf{v})\bar{q} = q\bar{q} + \epsilon q \mathbf{v} \bar{q} = 1 + \epsilon q \mathbf{v} \bar{q}$$

Translation as dual quaternion

- Idea: represent translation as a d.q. with unit ordinary part

$$\hat{t} = 1 + \epsilon \mathbf{x}$$

$$\mathbf{v} \mapsto \hat{t}\mathbf{v}\overline{\hat{t}^*} = (1 + \epsilon\mathbf{x})\mathbf{v}(1 + \epsilon\mathbf{x}) = \mathbf{v} + 2\epsilon\mathbf{x}$$

- This works but translates by 2**x** so we represent a translation by **w** as

$$\hat{t}(\mathbf{w}) = 1 + \frac{\epsilon\mathbf{w}}{2}$$

  – is this still a unit d.q.? yes: unit ordinary part; ordinary and dual parts orthogonal.

Rigid motion as dual quaternion

- Given a translation

$$T(\mathbf{v}) = R\mathbf{v} + \mathbf{w}$$

represent *R* and **w** as dual quaternions

$$q = q(R)$$

$$\hat{t} = \hat{t}(\mathbf{w})$$

and T is represented by the product $\hat{t}q$

$$(\hat{t}q)\mathbf{v}\overline{(\hat{t}q)^*}$$

$$\hat{t}q\mathbf{v}\overline{q^*}\overline{\hat{t}^*}$$

$$\hat{t}(q\mathbf{v}\overline{q^*})\overline{\hat{t}^*}$$

transforming by this product is the same as transforming by q, then t

Blending dual quaternions

- A generalization of slerp exists; it corresponds to a constant-speed screw motion from one location to the next
- As with quaternions, linear blending with renormalization provides a ready approximation
  - and it generalizes to more blending multiple dual quaternions
- This is a good way to blend rigid motions for skinning