COMP3271 Computer Graphics

Orientation Representation

2019-20

Objectives

Focus on the rotation transformation

Four orientation formats

- Rotation matrices
- Euler angles
- Axis-angle representation
- Quaternions

Comparisons of these representations

Criteria for Orientation Formats

How much storage is needed for the representation?

- How many numbers are needed to represent an orientation/rotation?

How efficient to form new orientations?

How efficient to rotate points and vectors?

How well the representation can be interpolated?

How suitable for numeric integration (e.g. for physical simulation)?

Rotation Matrices

$$R = \begin{pmatrix} u_0 & v_0 & w_0 \\ u_1 & v_1 & w_1 \\ u_2 & v_2 & w_2 \end{pmatrix}$$

The column vectors
$u = (u_0, u_1, u_2)^T, v = (v_0, v_1, v_2)^T,$
$w = (w_0, w_1, w_2)^T \in \mathbb{R}^3$
are three orthonormal basis vectors.

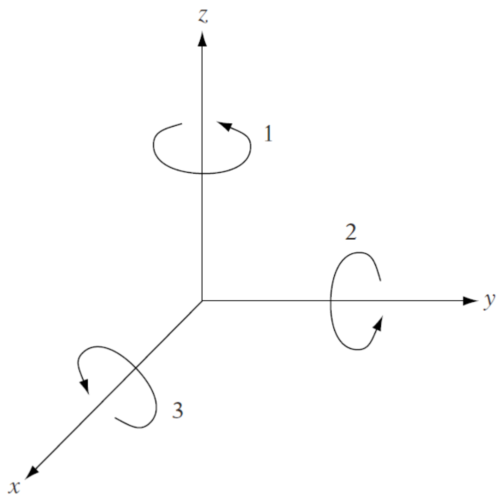
Nine numbers needed for a rotation

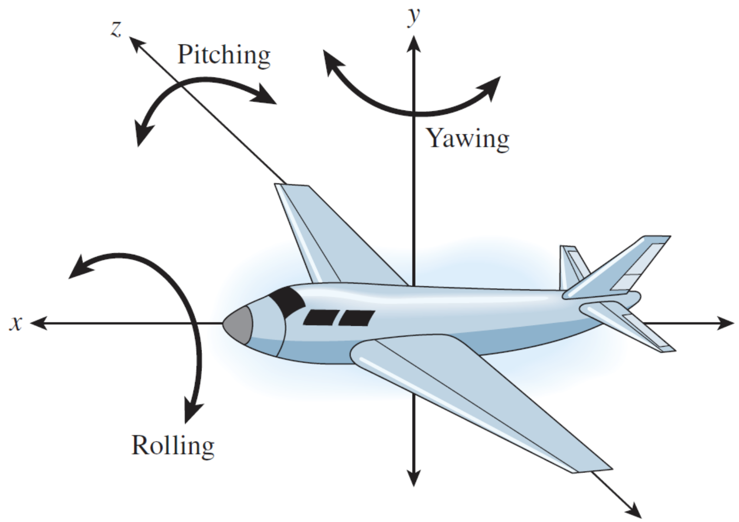
- Euler's rotation theorem states that we just need three numbers to represent a rotation

New rotations are obtained by matrix-matrix multiplication; vectors are rotated by matrix-vector multiplication

- Can be performed quite efficiently, some hardware has built-in circuitry for the multiplications

Euler Angles





Use 3 sequential rotations about a set of orthogonal axes to specify an orientation.

- If axes are fixed, need only 3 numbers for the angles (the Euler angles)
- If we choose the standard x-,y-,z-axes, the rotations are given by R_x, R_y, R_z
- No standard order for the use of the three axes

Composition of rotations and vector rotations resort to converting back to matrix representation and therefore are not efficient

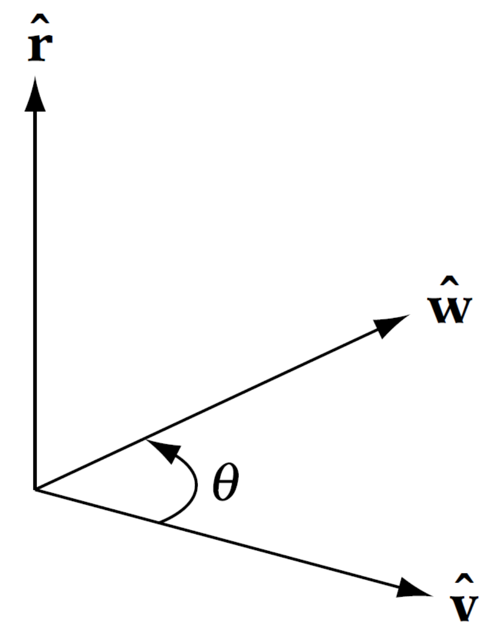
Axis-Angle Representation

Represent a rotation by an axis of rotation $\mathbf{r}$, and the angle of rotation θ about this axis

- $\mathbf{r}$ is normalized so the degree of freedom is 3

The axis-angle rotation to bring a vector $\mathbf{v}$ to another vector $\mathbf{w}$ is given by

$$\mathbf{r} = \hat{\mathbf{v}} \times \hat{\mathbf{w}}$$
$$\theta = \arccos(\hat{\mathbf{v}} \cdot \hat{\mathbf{w}})$$



Composition of rotations and vector rotations are not trivial.

Quaternions

Mathematical object developed by Sir William Rowan Hamilton in 1843 as an extension to the complex numbers

General form of a quaternion:

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

where $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ are "complex" numbers such that
$\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1$

A **quaternion** can therefore be represented as a 4-dimensional vector

$$\mathbf{q} = (w,\ x,\ y,\ z)$$

Quaternions

The $x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}$ part is similar to a 3D vector, so we may express a quaternion as

$$\mathbf{q} = (\underbrace{w}_{\text{scalar}},\ \underbrace{\mathbf{v}}_{\text{3D vector}})$$

A vector is represented as a quaternion by setting the scalar part 0:

$$\mathbf{q}_{\mathbf{u}} = (0,\ \mathbf{u})$$

Quaternion Normalization

Magnitude:

$$\|\mathbf{q}\| = \sqrt{(w^2 + x^2 + y^2 + z^2)}$$

Normalization:

$$\hat{\mathbf{q}} = \frac{\mathbf{q}}{\|\mathbf{q}\|}$$

Unit Quaternions as Rotations

A unit quaternion is a quaternion $\mathbf{q} = (w, \mathbf{v})$ such that

$$w^2 + \mathbf{v} \cdot \mathbf{v} = 1$$

$\mathbf{q}$ can also be written as

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

$\mathbf{r}$ is a unit vector representing the axis of rotation

θ is the angle of rotation

Example

What is the quaternion representing a rotation about the z-axis by 90 degrees?

$$w = \cos\left(\frac{\pi}{4}\right) = \frac{\sqrt{2}}{2}$$

$$x = 0 \cdot \sin\left(\frac{\pi}{4}\right) = 0$$

$$y = 0 \cdot \sin\left(\frac{\pi}{4}\right) = 0$$

$$\mathbf{q} = \left(\frac{\sqrt{2}}{2}, 0, 0, \frac{\sqrt{2}}{2}\right)$$

$$z = 1 \cdot \sin\left(\frac{\pi}{4}\right) = \frac{\sqrt{2}}{2}$$

Quaternion Operations

For addition and scalar multiplication, a quaternion behaves like a 4-vector:

$$
\begin{aligned}
(w_1, x_1, y_1, z_1) + (w_2, x_2, y_2, z_2) \\
= (w_1 + w_2, x_1 + x_2, y_1 + y_2, z_1 + z_2)
\end{aligned}
$$

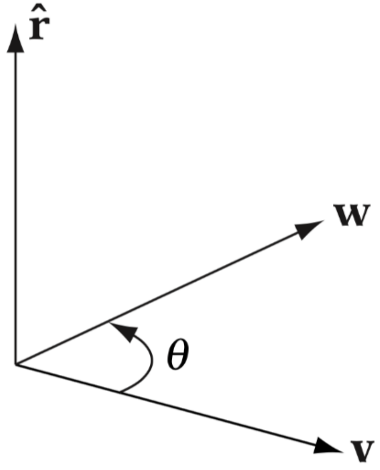
$$a(w, x, y, z) = (aw, ax, ay, az)$$

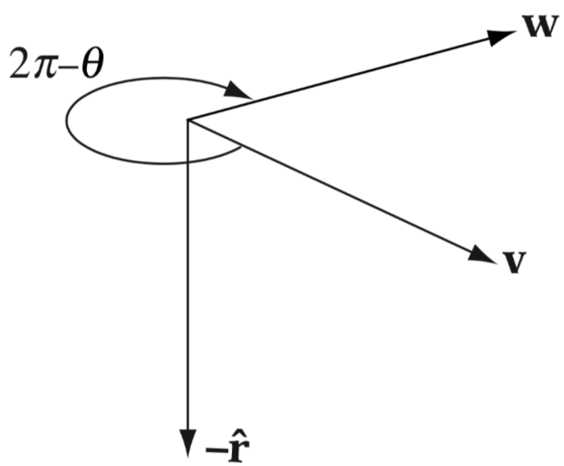
Given a quaternion $\mathbf{q}$, what is $-\mathbf{q}$?

Quaternion Negation

$$\mathbf{q} = (\cos\tfrac{\theta}{2}, \sin\tfrac{\theta}{2}\mathbf{r})$$

$$\begin{aligned}-\mathbf{q} &= (-\cos\tfrac{\theta}{2}, -\sin\tfrac{\theta}{2}\mathbf{r}) \\ &= (\cos\tfrac{2\pi+\theta}{2}, \sin\tfrac{2\pi+\theta}{2}\mathbf{r}) \\ &= (\cos\tfrac{2\pi-\theta}{2}, \sin\tfrac{2\pi-\theta}{2}(-\mathbf{r}))\end{aligned}$$





Quaternion Composition

Let $\mathbf{q}_1$ and $\mathbf{q}_2$ be two unit quaternions representing two rotations.

$$\mathbf{q}_1 = (w_1, \mathbf{v}_1) \qquad \mathbf{q}_2 = (w_2, \mathbf{v}_2)$$

The composition of first a rotation by $\mathbf{q}_1$ and then a rotation by $\mathbf{q}_2$ is given by the multiplication of $\mathbf{q}_2$ and $\mathbf{q}_1$:

$$\mathbf{q}_2\mathbf{q}_1 = (w_1 w_2 - \mathbf{v}_1 \cdot \mathbf{v}_2,\ w_1\mathbf{v}_2 + w_2\mathbf{v}_1 + \mathbf{v}_2 \times \mathbf{v}_1)$$

Order matters!

Vector dot product

Vector cross product

Compositing two rotations using quaternions take 16 multiplications and 12 additions.

Quaternion Inverse

The inverse of a quaternion $\mathbf{q}$ is denoted by $\mathbf{q}^{-1}$, such that

$$\mathbf{q}\mathbf{q}^{-1} = (1, 0, 0, 0)$$

Identity quaternion,
also representing zero rotation

Given $\mathbf{q} = (w, \mathbf{v})$, what is $\mathbf{q}^{-1}$?

$$\mathbf{q}^{-1} = (w, -\mathbf{v})$$

Negating the axis of rotation

Inverting a quaternion is fast!

Rotating Vectors with Quaternions

Let v be a quaternion representing a vector (x, y, z):

$$\mathbf{v} = (0, x, y, z)$$

Rotating a vector **v** by a unit quaternion **q** is done by:

$$\mathbf{v}' = \mathbf{q}\mathbf{v}\mathbf{q}^{-1}$$

Further apply a rotation by a unit quaternion **p**:

$$\mathbf{v}'' = \mathbf{p}\mathbf{q}\mathbf{v}\mathbf{q}^{-1}\mathbf{p}^{-1} = \mathbf{p}\mathbf{q}\mathbf{v}(\mathbf{p}\mathbf{q})^{-1}$$

$\mathbf{pq}$ is the composite rotation

Quaternion to Rotational Matrix

Let $\mathbf{q} = (w, x, y, z)$ be a unit quaternion
By simplifying the multiplications $\mathbf{q}\mathbf{v}\mathbf{q}^{-1}$
we can find that the rotation matrix corresponding to $\mathbf{q}$
is

$$R = \begin{pmatrix} 1-2y^2-2z^2 & 2xy-2wz & 2xz+2wy \\ 2xy+2wz & 1-2x^2-2z^2 & 2yz-2wx \\ 2xz-2wy & 2yz+2wx & 1-2x^2-2y^2 \end{pmatrix}$$

There are also fast methods for converting from a rotational matrix back to a quaternion, see
http://www.euclideanspace.com/maths/geometry/rotations/conversions/matrixToQuaternion/index.htm

Interpolating Rotations

A linear interpolation (LERP) from point $\mathbf{p}_1$ to point $\mathbf{p}_2$ is given by

$$(1-t)\mathbf{p}_1 + t\mathbf{p}_2,\ t \in [0,1]$$

How about interpolating two rotations R_1 and R_2?

Suppose R_1 and R_2 are represented by two matrices $\mathbf{M}_1$ and $\mathbf{M}_2$, can the following expression gives a proper linear interpolation?

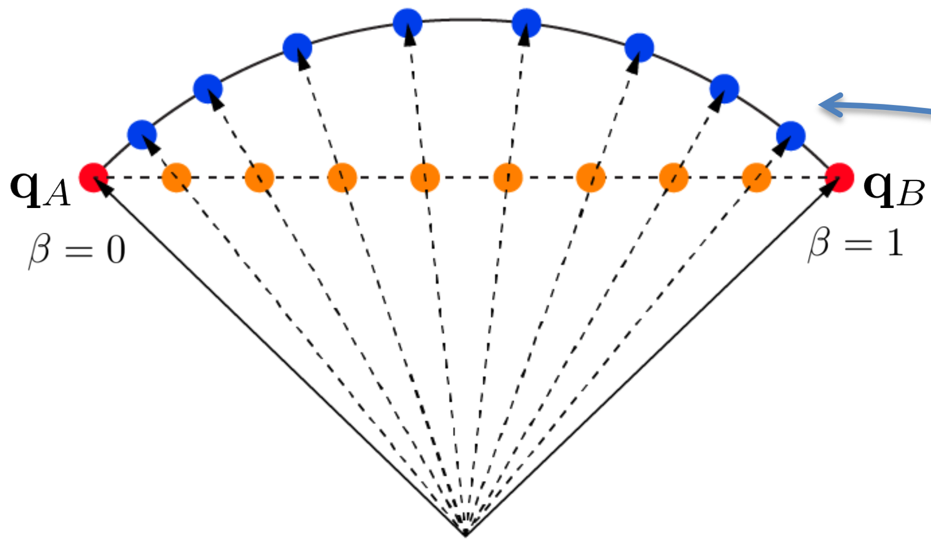
$$(1-t)\mathbf{M}_1 + t\mathbf{M}_2,\ t \in [0,1]$$

NO!

Quaternion Interpolations

We may apply a LERP (linear interpolation) to two quaternions to obtain the in-between rotations.

$$\mathbf{q}_{\mathrm{LERP}} = \mathrm{LERP}(\mathbf{q}_A, \mathbf{q}_B, \beta) = \frac{(1-\beta)\mathbf{q}_A + \beta\mathbf{q}_B}{|(1-\beta)\mathbf{q}_A + \beta\mathbf{q}_B|}$$



Angular speed of the rotational change is not constant. Slower near the end and faster at the middle.

Quaternion Interpolations

Spherical Linear Interpolation (SLERP)

- Interpolate along a great arc
- Constant angular speed in rotational change

$$\mathrm{SLERP}(\mathbf{q}_A, \mathbf{q}_B, \beta) = w_A \mathbf{q}_A + w_B \mathbf{q}_B$$

where

$$w_A = \frac{\sin((1-\beta)\theta)}{\sin\theta}$$

$$w_B = \frac{\sin(\beta\theta)}{\sin\theta}$$

ratio of arc lengths

$$\theta = \cos^{-1}(\mathbf{q}_A \cdot \mathbf{q}_B)$$

angle between $\mathbf{q}_A$ and $\mathbf{q}_B$