Interpolating Rotations - Slerps.

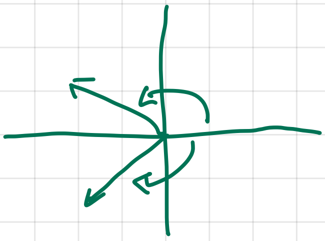
Idea: Interpolate using quaterions.

Problem: Interpolating rotation is tricker than positions size, color, etc.

- → Interpolating matrices doesn't make sense
- → Interpolating euler angles is problematic
  - → Don't always take "shortest routine"
    - Ex| Consider z rotation -170 version z rotation 160.
- → Need to be careful of special cases.

Aside:
How do we us linear interpolation to interpolate colors?

$$c = \overset{c_1}{\begin{pmatrix} r_1 \\ g_1 \\ b_1 \end{pmatrix}} (1-t) + \overset{c_2}{\begin{pmatrix} r_2 \\ g_2 \\ b_2 \end{pmatrix}} t$$



Solution: Use quaterions

"Method 1" Quick and Dirty

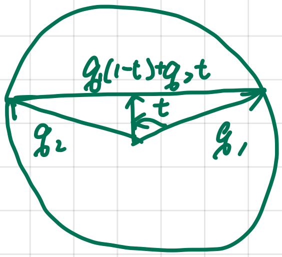
→ Linearly interpolate q_1, q_2, w/t.

$$q = LERP(q_1, q_2, t)$$

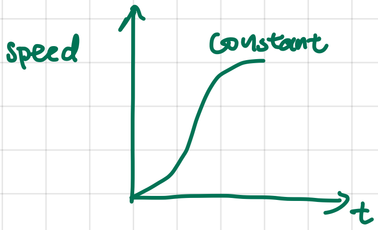
$$= q_1(1-t) + q_2 t, \quad t \in [0,1]$$

Normalize after to ensure length=1. ⇒ $q = \frac{\bar{q}}{\|q\|}$

Intuition:



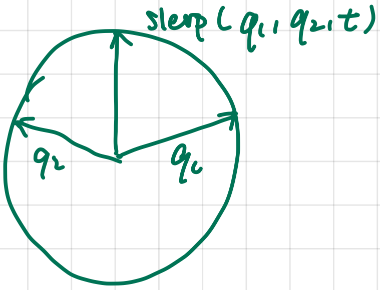
Potential Drawback:
speed from q_1 to q_2 is not constant



Method 2. Slerp

$$q = \frac{\sin(\Omega(1-t))}{\sin\Omega} q_1 + \frac{\sin(\Omega t)}{\sin\Omega} q_2, \quad \text{where } \cos\Omega = q_1 \cdot q_2$$

↑ preserve without length → intuition



Implementation Gotchas

① Potential divide by ∅

→ Return with q_1 or q_2

②. Potential NaN

when $\text{acos}(q_1, q_2) \in [-1, 1]$.

↑ float point impension

Use clamp

Ex Suppose

vector part

$q_1 = [\sin(0)\hat{k}, \cos(0)]$ ← rotation of 0 degrees.

w part

$q_2 = [\sin(\frac{90}{2})\hat{k}, \cos(\frac{90}{2})]$ ← 90 degrees around z axis.

$\hat{k} = (0, 0, 1)$

Compuee $q = \text{Slerp}(q_1, q_2, \frac{1}{2})$

Step 1: What are q_1 & q_2 as 4-tuples?

$$q_1 = (0, 0, 0, 1)$$

$$q_2 = (0, 0, \tfrac{\sqrt{2}}{2}, \tfrac{\sqrt{2}}{2})$$

Step 2: Compute the angle Ω

$$\cos\Omega = \frac{\sqrt{2}}{2}$$

$$\Omega = \frac{90}{2} = 45.$$

Step 3: Compute $q = \dfrac{\sin(\Omega(1-t))}{\sin\Omega} q_1 + \dfrac{\sin(\Omega t)}{\sin\Omega} q_2$

$$q = \frac{\sin(45(1-t))}{\sin 45} q_1 + \frac{\sin(45t)}{\sin 45} q_2$$

$$= \sqrt{2}\sin(45(1-t))q_1 + \sqrt{2}\sin(45t)q_2$$

$$= \sqrt{2}(\sin(45-45t))q_1 + \sqrt{2}\sin(45t)q_2$$

$$= \sqrt{2}\sin 22.5\, q_1 + \sqrt{2}\sin 22.5\, q_2$$

$$= \sqrt{2}\sin 22.5(q_1 + q_2)$$

$$= \sqrt{2}\sin 22.5\left(\begin{pmatrix}0\\0\\0\\1\end{pmatrix} + \begin{pmatrix}0\\0\\\frac{\sqrt{2}}{2}\\\frac{\sqrt{2}}{2}\end{pmatrix}\right)$$

$$= [0 \quad 0 \quad 0.3827 \quad 0.9239]$$

Check: $\text{acos}(0.9239) * 2 \approx 45°$

$\sin(\frac{45}{2}) \approx 0.3827$

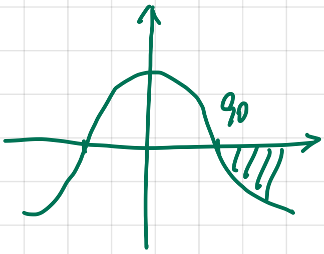
$(0.9239^2 + 0.3827^2)^{1/2} \approx 1$

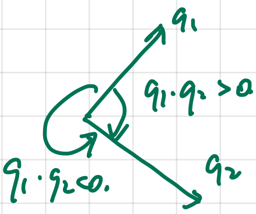
Guaranteeing the shortest path

we can rotate clock-wise or counter clockwise, which to choose?

Idea: If $q_1 \cdot q_2 < 0$. then the angle is "big"
i.e. q_1 and q_2. are far apart

why?





Take one of q_1 & q_2 and negate it

* They will have the same rotation.

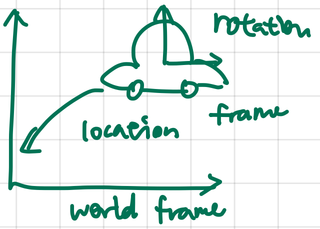
& interpolate

Frame of Reference

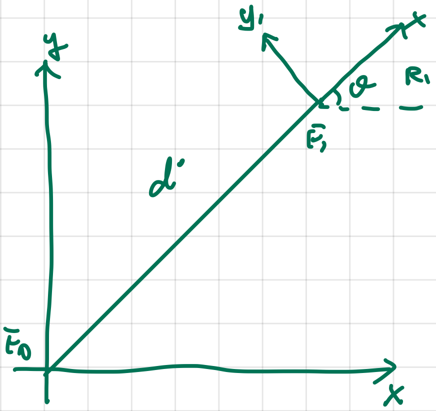
Recall: Candidates in space only make sense with a coordinate system.

The location and rotation of coordinates axis determine a frame of reference

Ex]



Ex]



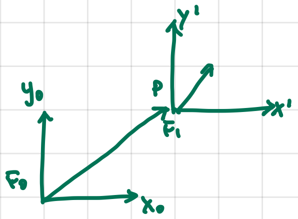
Let d_i^0 be the displacement from frame 1 to fram 0
Let R_i^0 be the Rotation from F_1 to F_0

Notation X_i^j
j ← to
i ← from

world frame location x,y,z direction
of the main coord. system
(aka global coordinates)

Ex Suppose $d_1^0 = (5,2,0)^T$

$R_1^0 = I$



Consider the origin of F_1 & call it p

$p' = (0,0,0)^T$ ← p wr.t. F_1

$p^0 = (5,2,0)^T$ ← p wr.t F_0

Homogeneous Coordinates

→ coordinates in "projective geometry"

Idea: Use 4-tuples / 4-D vectors to represent points and directions

→ A point $p = (p_x, p_y, p_z)^T$ has a homogeneous coordinate

$(p_x, p_y, p_z, 1)$

↑ 1 is 4th coordinate

→ A direction $v = (v_x, v_y, v_z)^T$ has homogenous coordinate

$(V_x, V_y, V_z, 0)^T$

↑ 4th coordinate is 0.

Transformations:

4 × 4 matrices that operate on homogeneous coordinates.

Ex Pure translation transform looks like

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ \hline 0 & 0 & 1 & 1 \end{array}\right] \left[\begin{array}{c} P_x \\ P_y \\ P_z \\ \hline 1 \end{array}\right] = \left[\begin{array}{c} P_x + a \\ P_y + b \\ P_z + c \\ \hline 1 \end{array}\right]$$

↑ multiplying a point

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ \hline 0 & 0 & 1 & 1 \end{array}\right] \left[\begin{array}{c} V_x \\ V_y \\ V_z \\ \hline 0 \end{array}\right] = \left[\begin{array}{c} V_x + 0.a \\ V_y + 0.b \\ V_z + 0c \\ \hline 0 \end{array}\right] = \left[\begin{array}{c} V_x \\ V_y \\ V_z \\ \hline 0 \end{array}\right]$$

↑ multiplying a vector direction does not change

Ex] Pure Rotation transform ($R_z(\theta)$)

$$\left[\begin{array}{ccc|c} c\theta & -s\theta & 0 & 0 \\ s\theta & c\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

- pt / Vector both going to change

We use "block matrix" notation to make working w/ transforms easier

"Pure" translation.

$$\left[\begin{array}{c|c} I & d \\ \hline 0 & 1 \end{array}\right] \rightarrow \text{3x1 block} \begin{pmatrix} dx \\ dy \\ dz \end{pmatrix}$$

3x3 block

1x3 block all 0's

1x1 block = 1

"Pure" Rotation

$$\left[\begin{array}{c|c} R & \emptyset \\ \hline 0 & 1 \end{array}\right]$$

Block notation for homogeneous coordinates.

$$\left[\begin{array}{c} P \\ \hline 1 \end{array}\right] \quad 3\times1 \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix} \qquad \left[\begin{array}{c} V \\ \hline 0 \end{array}\right] \begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix}$$

1x1

point