Representing Orientation
Representing Translations and Positions

• to translate by 30 units in $x$:
  – *add together thirty 1 unit translations*

• arithmetic interpolation
  (divide the total translation by $n$)
Representing Rotations and Orientations

• to rotate by 30 degrees:
  – $R' = R^{30}$
  ‣ where $R$ is a 3x3 or 4x4 matrix that rotates by one degree

• geometric interpolation
  (take the nth root of the desired final rotation matrix)
Representing Rotations and Orientations

• how many degrees of freedom in 3D?

• desired features of any representation
  – unique
  – continuous
  – compact
  – efficient to work with
Rotation in a 2D world
Rotation in a 3D world

- SO(3) group in Lie algebra

- four common alternative numerical representations:
  - 3x3 rotation matrix
  - Euler angles (fixed angles)
  - exponential map
  - unit quaternions
3x3 Rotation Matrix

- 9 elements
- 3 orthogonality constraints
- renormalization algorithms
- extracting pure rotational component (polar decomp)

\[
R = \begin{bmatrix}
  m_{11} & m_{12} & m_{13} \\
  m_{21} & m_{22} & m_{23} \\
  m_{31} & m_{32} & m_{33}
\end{bmatrix}
\]

\[
R^{-1} = R^T
\]

\[
\begin{align*}
  a \cdot b &= 0 & |a| &= 1 \\
  b \cdot c &= 0 & |b| &= 1 \\
  a \cdot c &= 0 & |c| &= 1
\end{align*}
\]

... and determinant = 1
Euler Angles

• choose 3 successive rotations about different axes
  – e.g., RPY: $z, y, x$

\[ R_{\text{RPY}} = \text{Rot}(z, \alpha) \text{Rot}(y, \beta) \text{Rot}(x, \gamma) \]

• common alternative: $z, x, z$
• problem: “gimbal lock”
• problem: non-uniqueness $\text{RPY}(0, 90, 0) = \text{RPY}(90, 90, 90)$
Euler’s Rotation Theorem

- can always go from one orientation to another with one rotation about a single axis

\[
\text{Rot} (\vec{k}, \theta) = \begin{bmatrix}
    k_x^2 v + c & k_x k_y v - k_z s & k_x k_z v + k_y s \\
    k_x k_y v + k_z s & k_y^2 v + c & k_y k_z v - k_x s \\
    k_x k_z v - k_y s & k_y k_z v + k_x s & k_z^2 v + c
\end{bmatrix}
\]

where
\[
\begin{align*}
    c &= \cos \theta \\
    v &= 1 - \cos \theta \\
    s &= \sin \theta
\end{align*}
\]
Exponential Map

- idea: encode amount of rotation into magnitude of $\vec{k}$
  \[|\vec{k}| = \theta \quad \text{Rot}(\vec{k}, |\vec{k}|) \quad \mathbb{R}^3 \rightarrow SO(3)\]
- axis definition undefined for no rotation
  - therefor define the zero vector to be the identity rotation
- singularities for $|\vec{k}| = 2\pi n$
Unit quaternions

\[ q = w + xi + yj + zk \]

\[
\begin{bmatrix}
  x \\
  y \\
  z \\
  w
\end{bmatrix} = (s, \bar{v})
\]

where \( q = (\cos \frac{\theta}{2}, \sin \frac{\theta}{2} \bar{k}) \)

- rotation of a vector, i.e., a point in a coord frame:
  \[ \bar{v}' = Rot(\bar{k}, \theta) \bar{v} = q \cdot \bar{v} \cdot \bar{q} \]
  \[ \bar{v} = (0, \bar{v}) \quad \bar{q} = (s, -\bar{v}) \]
- two successive rotations
  \[ q_2 (q_1 \cdot \bar{v} \cdot \bar{q}_1) \bar{q}_2 \]
Quaternion Math

\[ i^2 = -1 \quad i \cdot j = -j \cdot i = k \]
\[ j^2 = -1 \quad j \cdot k = -k \cdot j = i \]
\[ k^2 = -1 \quad k \cdot i = -i \cdot k = j \]

RH rule

\[ q^{-1} = \frac{1}{\|q\|^2} [s, -v] \]
\[ qq^{-1} = [1, (0,0,0)] \]

- unit quaternions

\[ w^2 + x^2 + y^2 + z^2 = 1 \]

- addition

\[ (s_1, v_1) + (s_2, v_2) = (s_1 + s_2, v_1 + v_2) \]

- multiplication

\[ (s_1, v_1) \cdot (s_2, v_2) = (s_1 \cdot s_2 - v_1 \cdot v_2, s_1 \cdot v_1 + s_2 \cdot v_2 + v_1 \times v_2) \]
Orientation Interpolation

- linear interpolation of quaternions
- note: $q$ and $-q$ represent the same orientation

$$q_1 \rightarrow q_2 \quad \text{or} \quad q_1 \rightarrow -q_2$$

choose shorter path, use dot product to compute

$$\cos \theta = q_1 \cdot q_2 = s_1 s_2 + v_1 \cdot v_2$$
Orientation Interpolation

**SLERP instead of LERP**

\[
slerp(q_1, q_2, u) = \frac{\sin((1-u)\theta)}{\sin \theta} q_1 + \frac{\sin(u\theta)}{\sin \theta} q_2
\]

smooth interpolation of multiple orientations:
- construct smooth curve on the 4D sphere