- Last Time:
  - SQP
  - Direct Collocation
- This Time:
  - Direct vs. DDP/iLQR
  - Dealing with Attitude (Rotation matrices + quaternions)

---

* DIRCOL vs. DDP

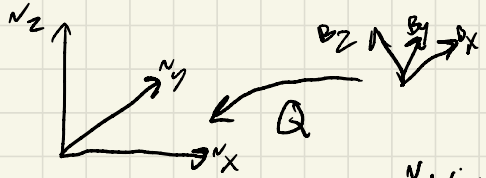
DIRCOL	DDP
Only respects dynamics at convergence	Always dynamically feasible
Can supply infeasible guess	Can only guess controls
Can handle arbitrary constraints	Hard to handle state constraints
Tracking controller must be designed seperately	TVLQR tracking controller is free
Typically not as fast	Very fast (local) convergence
Difficult to implement large-scale SQP solver	Easy to implement on embedded systems
Numerically robust	Has issues with ill-conditioning

- DDP is often a good choice for online/real-time applications where speed is critical and state constraints are not critical.

- DIRCOL is often a good choice for offline trajectory planning especially over long horizons and with complicated constraints.

Attitude

- Many robotic systems undergo large rotations (quadrotors, airplanes, spacecraft, underwater vehicles, legged robots.)
- Naive angle-based parameterizations (Euler angles) have singularities that cause failures and/or require hacks.
- Rotation matrices and quaternions are singularity free but optimizing over them requires some extra machinery.
- What is attitude?



rotation from body frame to inertial frame

$${}^{N}V = Q\,{}^{B}V$$

- 3 DOF, but there is no globally nonsingular 3-parameter attitude representation.

- Rotation / "Direction-Cosine" Matrix:

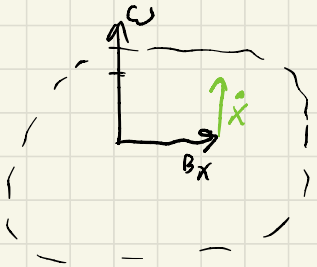
$$\begin{bmatrix} {}^N x_1 \\ {}^N x_2 \\ {}^N x_3 \end{bmatrix} = \underbrace{\begin{bmatrix} {}^B n_1^T \\ {}^B n_2^T \\ {}^B n_3^T \end{bmatrix}}_{Q} \begin{bmatrix} {}^B x_1 \\ {}^B x_2 \\ {}^B x_3 \end{bmatrix} = \underbrace{\begin{bmatrix} {}^N b_1 & {}^N b_2 & {}^N b_3 \end{bmatrix}}_{Q} \begin{bmatrix} {}^B x_1 \\ {}^B x_2 \\ {}^B x_3 \end{bmatrix}$$

$$\Rightarrow Q^T Q = I \Rightarrow Q^{-1} = Q^T \quad \text{"orthogonal"}$$

$$\Rightarrow \det(Q) = 1 \quad \text{"special"}$$

$Q \in SO(3)$ "Special orthogonal group in 3D"

- Kinematics (how do we integrate angular velocity)



$${}^N x = Q\,{}^B x, \quad {}^N \dot{x} = {}^N\omega \times {}^N x$$

$$= Q({}^B\omega \times {}^B x)$$

$$\omega \times x = \underbrace{\begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}}_{\hat{\omega}} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

- Apply chain rule:

$${}^N x = Q\,{}^B x \Rightarrow {}^N \dot{x} = \dot{Q}\,{}^B x + Q\,{}^B \dot{x}^{\,\nearrow 0}$$

$$\Rightarrow \dot{Q}\,{}^B x = Q({}^B\omega \times {}^B x) = Q\,\hat{\omega}\,{}^B x \Rightarrow \boxed{\dot{Q} = Q\,{}^B\hat{\omega}}$$

- Now we could do dynamics with a rotation matrix in our state. but has a lot of redundancy and suffers from constraints.

- Quaternions are more compact / computationally efficient.

- Define axis of rotation (unit vector) a

- Define angle of rotation (scalar, radians) θ

$$\underbrace{\phi}_{\text{"axis-angle" vector} \in \mathbb{R}^3} = a\theta, \quad \|\phi\| = \theta, \quad \frac{\phi}{\|\phi\|} = a$$

- For a constant ω (or very short h) can think of ϕ as integral of ω:

$$\phi \approx \int_0^h \omega \, dt$$

(not true in general)

- In terms of axis-angle, we can define quaternion:

$$q = \begin{bmatrix} \cos(\theta/2) \\ a \sin(\theta/2) \end{bmatrix} \begin{matrix} \leftarrow \text{"scalar part"} \\ \leftarrow \text{"vector part"} \end{matrix}$$

* $q^T q = 1 \Rightarrow$ valid rotations rotations correspond to unit quaternions. Easy to normalize

* q and $-q$ correspond to the same rotation (add 2π to θ). Called "double cover"

* Operations on quaternions are analogous to rotation matrices.

- Quaternion Multiplication:

$$q_1 * q_2 = \begin{bmatrix} s_1 \\ v_1 \end{bmatrix} * \begin{bmatrix} s_2 \\ v_2 \end{bmatrix} = \begin{bmatrix} s_1 s_2 - v_1^T v_2 \\ s_1 v_2 + s_2 v_1 + v_1 \times v_2 \end{bmatrix}$$

$$= \underbrace{\begin{bmatrix} s_1 & -v_1^T \\ v_1 & s_1 I + \hat{v}_1 \end{bmatrix}}_{L(q_1)} \begin{bmatrix} s_2 \\ v_2 \end{bmatrix} = \underbrace{\begin{bmatrix} s_2 & -v_2^T \\ v_2 & s_2 I - \hat{v}_2 \end{bmatrix}}_{R(q_2)} \begin{bmatrix} s_1 \\ v_1 \end{bmatrix}$$

- Quaternion Identity:

$$\theta = 0 \Rightarrow \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

- Quaternion Conjugate

$$q^+ = \begin{bmatrix} s \\ -v \end{bmatrix} = Tq = \begin{bmatrix} 1 & 0 \\ 0 & -I \end{bmatrix} \begin{bmatrix} s \\ v \end{bmatrix}$$

- Rotating a Vector:

$$\underbrace{\begin{bmatrix} 0 \\ {}^N x \end{bmatrix}}_{{}^N\hat{x}} = q * \begin{bmatrix} 0 \\ {}^B x \end{bmatrix} * q^+ = \underbrace{L(q) R(q)^T}_{\text{gives } Q(q)} {}^B x = R_{(q)}^T L(q) {}^B x$$

$$\underbrace{H}_{\begin{bmatrix} 0 \\ I \end{bmatrix}} x = \begin{bmatrix} 0 \\ x \end{bmatrix}$$

- Quaternion Kinematics

$$\dot{q} = \frac{1}{2} q * \begin{bmatrix} 0 \\ \omega \end{bmatrix} = \underbrace{\frac{1}{2} L(q) H}_{4\times 3} \omega$$

- Now we can simulate dynamics with quaternions