Attitude Control Crash Course

- We're trying to point our satellite with only torque coils (no reaction wheels)
- This is hard because the system is underactuated

---

* Underactuated Systems:

- Manipulator Equation:

$$\underbrace{M(q)}_{\substack{\text{mass}\\ \text{matrix}}}\dot{v} + \underbrace{C(q,v)v}_{\substack{\text{coriolis}\\ \text{term}}} + \underbrace{G(q)}_{\substack{\text{potential}\\ \text{term}}} = \underbrace{B(q)}_{\substack{\text{Input}\\ \text{Jacobian}}}u$$

q = "configuration" or "generalized coordinates"

v = velocity

u = control input

- A system is fully actuated if $\text{rank}(B(q)) = \dim(v)$.
- If this is true, we can cancel out the natural dynamics and do whatever we want:

$$u = -\underbrace{B^{-1}}_{\text{must be invertable!}}(M\dot{v} + Cv + G)$$

* Satellite Dynamics:

$$\underbrace{J\dot{\omega} + \omega \times J\omega = \tau}_{\text{Euler's Equation}}, \quad \tau = \underbrace{m}_{\substack{\text{magnetic}\\\text{moment}}} \times \underbrace{b}_{\substack{\text{Earth's}\\\text{magnetic field}}} = -b \times m$$

- In manipulator form:

$$M(q) = J \quad , \quad C(q,v) = \hat{\omega} J \;, \quad B(q) = -\hat{b}$$

- Hat matrix:

$$\hat{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \Rightarrow \hat{\omega} x = \omega \times x$$

- Note $\text{rank}(\hat{b}) = 2 \Rightarrow$ underactuated!

Trajectory Optimization

- We're going to turn the control problem into the following optimization problem:

$$\underset{\substack{\underbrace{x_{1:N}}_{\text{states}},\ \underbrace{u_{1:N-1}}_{\text{controls}}}}{\text{minimize}} \quad \underbrace{J}_{\substack{\text{cost}\\\text{function}}} = \sum_{k=1}^{N-1} \underbrace{L(x_n, u_n)}_{\substack{\text{stage}\\\text{cost}}} + \underbrace{L_F(x_N)}_{\substack{\text{Terminal}\\\text{cost}}}$$

$$\text{s.t.} \quad x_{n+1} = f(x_n, u_n) \leftarrow \text{dynamics constraints}$$

$$u_{min} \le u_n \le u_{max} \leftarrow \text{input constraints}$$

- We're going to assume our cost functions are quadratic

$$L(x,u) = \frac{1}{2} x^T Q x + q^T x + \frac{1}{2} u^T R u + r^T u$$

$$L_F(x) = \frac{1}{2} x^T Q_F x + q_F^T x$$

- We will also need to linearize our dynamics:

$$x_{n+1} + \delta x_{n+1} = f(x_n + \delta x_n, u_n + \delta u_n) \approx f(x_n, u_n) + \underbrace{A_n}_{\frac{\partial f}{\partial x}} \delta x_n + \underbrace{B_n}_{\frac{\partial f}{\partial u}} \delta u_n$$

$$\Rightarrow \delta x_{n+1} = A_n \delta x_n + B_n \delta u_n$$

- If the dynamics were actually linear, this would be an LQR problem. We're going to iteratively linearize + solve LQR problems until convergence.

* LQR (Linear-Quadratic Regulator)

- We're going to calculate the solution backward from the goal.

- Define the "cost-to-go" function

$$V_k(x) = \frac{1}{2} x^T S_k x + s_k^T x$$

- Starting from the end:

$$V_N(x) = L_F(x) = \frac{1}{2} x^T Q_F x + q_F^T x$$

$$\Rightarrow S_N = Q_F \quad , \quad s_N = q_F$$

- Backing up one step:

$$V_{N-1}(x) = \min_{u_{N-1}} L(x_{N-1}, u_{N-1}) + V_N(A_{N-1} x_{N-1} + B_{N-1} u_{N-1})$$

$$\Rightarrow \frac{\partial L}{\partial u_{N-1}} + \frac{\partial V_N}{\partial x_N} \underbrace{\frac{\partial x_N}{\partial u_{N-1}}}_{B_{N-1}} = 0$$

$$\Rightarrow R u_{N-1} + r + B_{N-1}^T S_N (A x_{N-1} + B_{N-1} u_{N-1}) + B_{N-1}^T s_N = 0$$

$$\Rightarrow \boxed{\begin{aligned} u_{N-1} &= -(R + B_{N-1}^T S_N B_{N-1})^{-1}(r + B_{N-1}^T s_N + B_{N-1}^T S_N A_{N-1} x_{N-1}) \\ &= -l_{N-1} - K_{N-1} x_{N-1} \end{aligned}}$$

- Plug u_{N-1} back into cost-to-go:

$$V_{N-1}(x) = L(x_{N-1}, -l_{N-1} - K_{N-1} x_{N-1}) + V_{N-1}([A_{N-1} - B_{N-1} K_{N-1}] x_{N-1} - l_{N-1})$$

$$= \frac{1}{2} x^T S_{N-1} x + s_{N-1}^T x$$

$$\Rightarrow \boxed{\begin{aligned} S_{N-1} &= Q + K_{N-1}^T R K_{N-1} + (A_{N-1} - B_{N-1} K_{N-1})^T S_N (A_{N-1} - B_{N-1} K_{N-1}) \\ s_{N-1} &= q - K_{N-1}^T r + K_{N-1}^T R l_{N-1} + (A_{N-1} - B_{N-1} K_{N-1})^T (s_N - S_N l_{N-1}) \end{aligned}}$$

- Now we're back to where we started and we can continue backwards until we get to $k = 1$