16-745 Optimal Control Lecture 16

Reid Graves

March 18, 2025

1 Last Time

- Optimization with Quaternions

2 Today

- LQR with Quaternions
- Quadrotor Control

---

3 LQR with Quaternions

- Naively linearizing a system with a quaternion state results in an uncontrollable linear system
- We'll apply our quaternion differentiation tricks to LQR to make this work
- Given a reference $\bar{x}_k, \bar{u}_k$ for a discrete time system $f(x_k, u_k)$:

$$\begin{aligned}\bar{x}_{k+1} + \Delta x_{k+1} &\approx f(\bar{x}_k + \Delta x_k, \bar{u}_k + \Delta u_k) \\ &\text{First order Taylor expansion:} \\ &\approx f(\bar{x}_k, \bar{u_k}) + \underbrace{A_k}_{\frac{\partial f}{\partial x}} \Delta x_k + \underbrace{B_k}_{\frac{\partial f}{\partial u}} \Delta u_k \\ \bar{x}_{k+1} &= f(\bar{x}_k, \bar{u_k})\end{aligned}$$

$$\Delta x_{k+1} = \underbrace{A_k}_{\frac{\partial f}{\partial x}} \Delta x_k + \underbrace{B_k}_{\frac{\partial f}{\partial u}} \Delta u_k$$

- For the quaternion part of the state, we apply the attitude Jacobian to convert $\Delta \to \phi \in \mathbb{R}^3$:

$$x = \begin{bmatrix} r \\ q \\ \theta \\ r \\ \omega \\ \dot{\theta} \end{bmatrix}$$

$$\begin{aligned} r \quad & x[1:3] \\ q \quad & x[4:7] \\ \theta \quad & x[8:n] \text{ (joint angles)} \\ & \vdots \end{aligned}$$

$$\underbrace{\begin{bmatrix} \Delta x_{k+1}[1:3] \\ \phi_{k+1} \\ \Delta x_{k+1}[8:n] \end{bmatrix}}_{\Delta\tilde{x}_{k+1}} = \underbrace{\begin{bmatrix} I & & 0 \\ & G(\bar{q}_{k+1}) & \\ 0 & & I \end{bmatrix}^T}_{E(\bar{x}_{k+1})} A_k \underbrace{\begin{bmatrix} I & & 0 \\ & G(\bar{q}_k) & \\ 0 & & I \end{bmatrix}}_{E\bar{x}_k} \begin{bmatrix} \Delta x_k[1:3] \\ \phi_k \\ \Delta x_k[8:n] \end{bmatrix} + E^T(\bar{x}_{k+1}) B_k \Delta u_k$$

Since the B matrix is multiplied by controls, it doesn't need a Jacobian transform on the right. But since it's output is a state, it needs a Jacobian transform on the left.

- Once we have these "reduced" Jacobians $\tilde{A}_k, \tilde{B}_k$:

$$\tilde{A}_k = E(\bar{x}_{k+1})^T A_k E(\bar{x}_k) \qquad \tilde{B}_k = E^T(\bar{x}_{k+1}) B_k$$

We compute the LQR controller as usual.

- When we run the controller, we calculate $\Delta\tilde{x}$ before multiplying by K:

$$\text{given } x_k \quad \Delta\tilde{x}_k = \begin{bmatrix} x_k[1:3] - \bar{x}_k[1:3] \\ \phi(L(\bar{q}_k)^T q_k) \\ x_k[8:n] - \bar{x}_k[8:n] \end{bmatrix} \to \text{whatever 3 parameter representation you like}$$

$$u_k = \bar{u}_k - K_k \Delta\tilde{x}_k$$

3.1 Computing error/delta rotations

- Many possible conventions
- We will write it as rotation from body frame B to the reference/desired body frame R. We want to have the rotation from the body frame to the inertial frame:

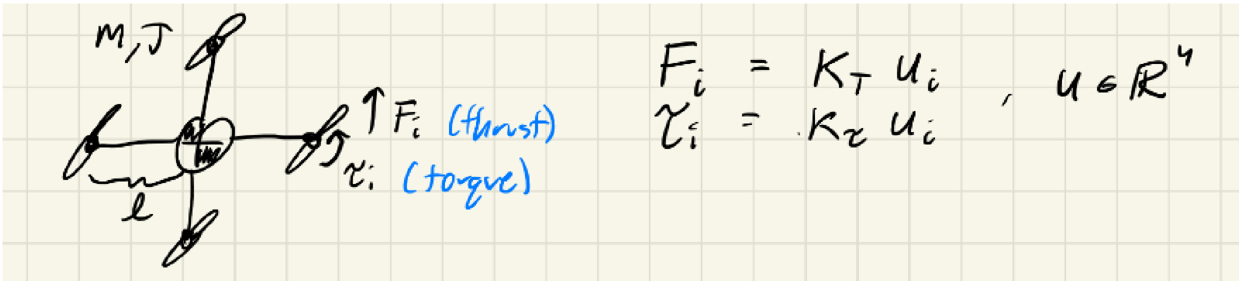
$$\Rightarrow \underbrace{{}^N Q^B}_{Q} = \underbrace{{}^N Q^R}_{\bar{Q}} \underbrace{{}^R Q^B}_{\Delta Q} \qquad \Rightarrow \underbrace{({}^N Q^R)^{-1} {}^N Q^B}_{\bar{Q}^T Q} = \underbrace{{}^R Q^B}_{\Delta Q}$$

Q is Inertial frame, $\bar{Q}$ is the frame we want, and ΔQ is the error

- Using quaternions

$$\Delta q = \bar{q}^\dagger * q = L(\bar{q})^T q$$

4 3D Quadrotor



- State:

$$x = \begin{bmatrix} {}^N r \in \mathbb{R}^3 & \text{Position in } N \text{ frame} \\ {}^N q^B \in \mathbb{H} & \text{attitude } B \to N \\ {}^B v \in \mathbb{R}^3 & \text{linear velocity in } B \text{ frame} \\ {}^B \omega \in \mathbb{R}^3 & \text{angular velocity in } B \text{ frame} \end{bmatrix}$$

- Kinematics

$$\begin{aligned} {}^N\dot{r} &= {}^N v = Q\ {}^B v \\ \dot{q} &= \frac{1}{2} q * \hat{w} = \frac{1}{2} L(q) H\ {}^B\omega = \frac{1}{2} G(q)\ {}^B\omega \end{aligned}$$

- Translation Dynamics

$$\underbrace{m\ {}^N\dot{v}}_{\text{need to rotate into } B \text{ frame}} = {}^N F \leftarrow \text{total force}$$

$$ {}^N v = Q\ {}^B v \Rightarrow {}^N \dot{v} = \underbrace{\dot{Q}}_{Q\hat{\omega}}\ {}^B v + Q\ {}^B\dot{v} = Q\hat{\omega}\ {}^B v + Q\ {}^B\dot{v}$$

$$\Rightarrow {}^B \dot{v} = \overbrace{Q^T\ {}^N\dot{v}}^{\text{rotate into } B \text{ frame}} - \underbrace{{}^B\omega \times {}^B v}_{\text{extra term from spin}}$$

$$\Rightarrow {}^B \dot{v} = \frac{1}{m}\ {}^B F - {}^B\omega \times {}^B v$$

$${}^B F = Q^T \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ k_T & k_T & k_T & k_T \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}$$

- Rotation Dynamics:

$$\underbrace{J\ {}^B\dot{\omega} + {}^B\omega + {}^B\omega \times J\ {}^B\omega = {}^B\tau}_{\text{“Eulers equation”}}$$

J : Inertia matrix

τ : total torque

$${}^B\tau = \begin{bmatrix} lk_T(u2 - u_4) \\ lk_T(u_3 - u_1) \\ k_\tau(u_1 - u_2 + u_3 - u_4) \end{bmatrix}$$

5 Example

- LQR (or convex MPC) with some quaternion tricks is very effective.