Last Time:
- Dynamic Programming
- Convex Optimization

Today:
- Convex MPC

---

* Convex MPC

- Think of this as "constrained LQR"
- Remember from DP, if we have the cost-to-go we can get u_k by solving a one-step problem:

$$u_k = \underset{u}{\operatorname{argmin}} \; l(x_k, u) + V_{k+1}(f(x_k, u))$$

$$= \underset{u}{\operatorname{argmin}} \; \frac{1}{2} u^T R_k u + (A_k x_k + B_k u)^T P_{k+1} (A_k x_k + B_k u)$$

- We can add constraints to this one-step problem, but performance will be bad because $V_k(x)$ was computed without constraints.
- There's no reason I can't add more steps to the one-step problem:

$$\min_{\substack{x_{1:H} \\ u_{1:H-1}}} \sum_{k=1}^{H-1} \frac{1}{2} x_k^T Q_k x_k + \frac{1}{2} u_k^T R_k u_k + x_H^T P_H x_H$$

$$\text{s.t.} \quad x_{k+1} = A_k x_k + B_k u_k$$

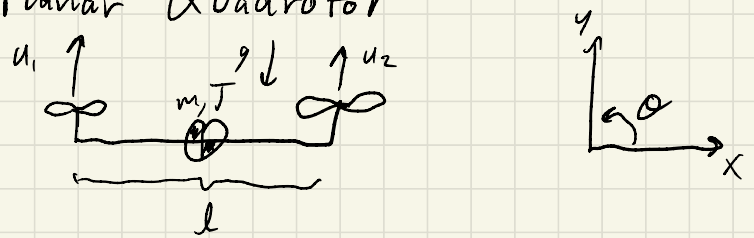
$$x_k \in \mathcal{X}, \quad \mathcal{X}, \mathcal{U} \text{ convex}$$

$$u_k \in \mathcal{U}$$

(e.g. $u_{min} \le u_k \le u_{max}$)

- $H \leq N$ is called the "horizon"
- With no additional constraints MPC / "receding horizon" exactly matches LQR for any H
- Intuition: explicit constrained optimization over first H steps gets the state close enough to the origin/reference so that constraints are no longer active and LQR solution is valid farther into the future.
- In general:
  - A good approximation of $V(x)$ is important for good performance
  - Better $V(x) \Rightarrow$ shorter horizon will work
  - Longer $H \Rightarrow$ better performance, less reliant on quality of $V(x)$ approximation

* Example
  - Planar Quadrotor



$$m\ddot{x} = -(u_1 + u_2)\sin(\theta)$$

$$m\ddot{y} = (u_1 + u_2)\cos(\theta) - mg$$

$$J\ddot{\theta} = \frac{l}{2}(u_2 - u_1)$$

- Linearize about hover:

$$\Rightarrow \quad u_1 = u_2 = \frac{1}{2}mg \quad \text{(cancel gravity)}$$

$$\Rightarrow \begin{cases} \Delta\ddot{x} \approx -g\theta \\ \Delta\ddot{y} \approx \frac{1}{m}(\Delta u_1 + \Delta u_2) \\ \Delta\ddot{\theta} \approx \frac{1}{J}\frac{l}{2}(\Delta u_2 - \Delta u_1) \end{cases}$$

$$\begin{bmatrix} \Delta\dot{x} \\ \Delta\dot{y} \\ \Delta\dot{\theta} \\ \Delta\ddot{x} \\ \Delta\ddot{y} \\ \Delta\ddot{\theta} \end{bmatrix} = \underbrace{\left[\begin{array}{ccc|ccc} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & -g & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]}_{A} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \\ \Delta\dot{x} \\ \Delta\dot{y} \\ \Delta\dot{\theta} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \hline 0 & 0 \\ 1/m & 1/m \\ -l/2J & l/2J \end{bmatrix}}_{B} \begin{bmatrix} \Delta u_1 \\ \Delta u_2 \end{bmatrix}$$

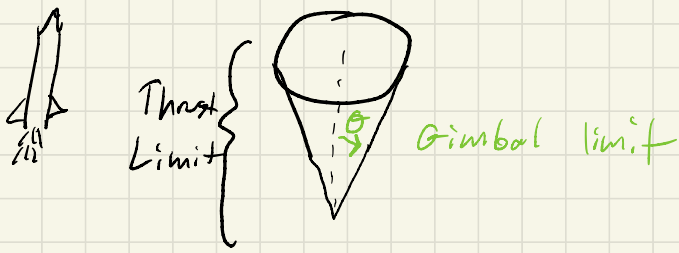
- MPC Cost Function:

$$J = \sum_{n=1}^{H} \frac{1}{2}(x_n - x_{ref})^T Q (x_n - x_{ref}) + \frac{1}{2}\Delta u_n^T R \Delta u_n + (x_H - x_{ref})^T P_H (x_n - x_{ref})$$

$$\frac{1}{2}x_n^T Q x_n - \underbrace{x_{ref}^T Q}_{q^T} x + \dots$$

* Other MPC Examples

- Rocket Landing
  - Thrust vector constraints are naturally expressed as a cone:

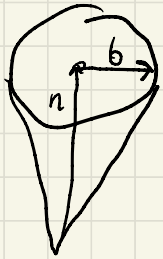


  - Mars powered descent and SpaceX use SOCP-based MPC with linearized dynamics

- Legged Locomotion
  - Contact forces must obey "friction cone" constraint

$$\|b\|_2 \leq \mu n$$

b: Friction Force; μ: Friction coefficient; n: normal force



  - Cone is often approximated as a pyramid so the constraint is linear ($Ax \leq b$)

- MIT Cheetah and other quadrupeds use QP-based MPC with linearized body dynamics + friction pyramid

* What about nonlinear dynamics?

- Linear stuff often works well, so use it if you can.

- Nonlinear dynamics are risky because optimization problems are non-convex

- No convergence guarantees

- Can work well in practice due to warm starting (solution doesn't change much over on iteration)

- Active area of research

- Common in autonomous driving