2/23 Recitation

- quiz 5 solutions
- Trajectory Optimization
    - iLQR/DDP vs Dircol
    - when to what

Quiz 5: Clamping (projecting) solutions

DO NOT DO THIS:

$$\min_x \frac{1}{2} x^T Q x + q^T x$$

$$\text{s.t.} \quad \underline{x} \le x \le \overline{x}$$

$$\downarrow$$

$$x^* = -Q^{-1} q$$ - unconstrained solution

$$\downarrow$$

$$\text{clamp}(x^*, \underline{x}, \overline{x})$$ project into feasible

$$\downarrow$$

suboptimal

If a problem has constraints, you cannot simply solve the unconstrained problem and project it on the constraints. If you do this, it will be suboptimal. This is why we can't simply clamp the LQR solution to an optimal control problem with control constraints. Try this for yourself and see.

Trajectory Optimization

Problem	Solver
IHLQR $\quad \min_{\substack{x_{1:\infty} \\ u_{1:\infty}}} \sum_{i=1}^{\infty} x_i^T Q x_i + u_i^T R u_i$ s.t. $\quad x_1 = x_{IC}$ $x_{i+1} = A x_i + B u_i$	Ricatti, and DARE solver for $u = -Kx$
TVLQR $\quad \min_{\substack{x_{1:N} \\ u_{1:N-1}}} \underbrace{\sum_{i=1}^{N-1} x_i^T Q x_i + u_i^T R u_i}_{\text{Stage cost}} + \underbrace{x_N^T Q_N x_N}_{\text{term. cost}}$ s.t. $\quad x_1 = x_{IC}$ $x_{i+1} = A_i x_i + B_i u_i$	Ricatti for $u_i = -K_i x_i$ CVX for x's, u's
CVX trajopt $\quad \min_{\substack{x_{1:N} \\ u_{1:N-1}}} \sum_{i=1}^{N-1} \ell(x_i, u_i) + \ell_N(x_N)$ Convex s.t. $\quad x_1 = x_{IC}$ $x_{i+1} = A_i x_i + B_i u_i$ $h_i(x_i, u_i) = 0$ Linear $g_i(x_i, u_i) \le 0$ Convex	any CVX solver that can handle the costs/constraints (LP, QP, SOCP)
nonlinear dynamics $\quad \min_{\substack{x_{1:N} \\ u_{1:N-1}}} \sum_{i=1}^{N-1} \ell_i(x_i, u_i) + \ell_N(x_N)$ don't have to be convex s.t. $\quad x_1 = x_{IC}$ $x_{i+1} = f(x_i, u_i)$ nonlinear	iLQR/DDP or DIRCOL

$$\min_{\substack{x_{1:N} \\ u_{1:N-1}}} \sum_{i=1}^{N-1} \ell_i(x_i, u_i) + \ell_N(x_N)$$

don't have to be convex

$$\text{s.t.} \quad x_1 = x_{IC}$$

$$x_{i+1} = f(x_i, u_i)$$ nonlinear

See DIRCOL section

$$h_i(x_i, u_i) = 0$$

not convex

$$g_i(x_i, u_i) \le 0$$

DIRCOL,

or iLQR/DDP + constraint handling
(ALTRO, iLQR + AL)

iLQR/DDP

+ easy to implement / easy debug
+ get a TVLQR controller for free
+ can be very fast
- constraints are hard, must use something like ALTRO
- Only initialize u's, so initial guess has to obey dynamics
+ always dynamically feasible

DIRCOL

- must use NLP solver (IPOPT, SNOPT), or custom sequential convex programming (SCP) method
+ initialize x's, u's, both, neither
+ explicit vs. implicit integrators
+ constraint handling
+ maybe more robust than iLQR/DDP + constraints
- slower than iLQR/DDP
+ ZOH, FOH, anything I want

DIRCOL

$$\min_{\substack{x_{1:N} \\ u_{1:N-1}}} \sum_{i=1}^{N-1} l_i(x_i, u_i) + l_N(x_N)$$

$$\text{s.t.} \quad x_1 = x_{IC}$$

either or

$$0 = f_{imp}(x_i, x_{i+1}, u_i) \quad \text{Implicit integrator}$$

$$0 = f_{exp}(x_i, u_i) - x_{i+1} \quad \text{explicit integrator}$$

$$h_i(x_i, u_i) = 0$$

$$g_i(x_i, u_i) \le 0$$

DIRCOL will work with either an implicit or explicit integrator. Since we are just writing down our dynamics constraints as equality constraints, we can either use the explicit integration style f(xi,ui) - x{i+1} = 0, or we can use the implicit integration residual f(xi,ui,x{i+1}) = 0. Since it doesn't matter to the solver which one we use, we may as well use the implicit one since it's a little more accurate for the same order.

how to use NLP solver

$$\min_z f(z)$$

$$\text{s.t.} \quad C(z) = 0$$

$$L \le h(z) \le U$$

$$\underline{z} \le z \le \overline{z}$$

$$z = \begin{bmatrix} x_1 \\ u_1 \\ x_2 \\ u_2 \\ \vdots \\ x_{N-1} \\ u_{N-1} \\ x_N \end{bmatrix}$$

In order to actually use an NLP solver like IPOPT or SNOPT, we have to formulate our problem such that we are solving for a single vector Z. We accomplish this by simply stacking up all of our x's and u's into a Z vector.