Last Time:

- How to land a space ship

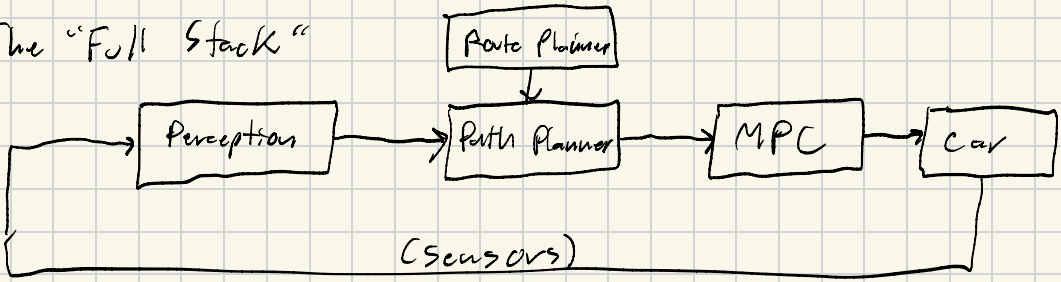
Today:

- How to drive a car
- Game-Theoretic MPC

---

* History
  - Primitive demos going back to 1920's-30's
  - First modern work in 1980's at CMU + Germany
  - Lots of demos in 1990's with 98% + autonomy on cross-country highway drives
  - DARPA challenge in early 2000's

* The "Full Stack"



- Sensors: GPS, IMUs, Cameras, RADAR, LIDAR
- Perception + human-driver prediction are the hard parts
- Route planning: graph search to generate waypoints

- Path Planner: Generate a smooth spline curve that is collision free (no dynamics)

- MPC Controller: Tracks spline curve while reasoning about vehicle dynamics + constraints.

Vehicle Dynamics

- Lots of options with different levels of fidelity
- Most Common: "bicycle" or "single-track" models
- Kinematic Bicycle Model:

* Inputs: V, α (or $\dot{V}$, $\dot{\alpha}$)

* Assume tires don't slip:

$$\begin{aligned} \dot{x} &= V\cos(\theta) \\ \dot{y} &= V\sin(\theta) \\ \dot{\theta} &= \frac{V\tan(\alpha)}{l} \\ \dot{V} &= u_1 \\ \dot{\alpha} &= u_2 \end{aligned} \quad , \quad x = \begin{bmatrix} x \\ y \\ \theta \\ V \\ \alpha \end{bmatrix} \quad u = \begin{bmatrix} \dot{V} \\ \dot{\alpha} \end{bmatrix}$$

- Works well "normal" driving
- Breaks down in more aggressive settings (high acceleration)

- More Complex:
  - "Dynamic" Bicycle model: Model car as a rigid body, $F = ma$, $\tau = J\dot{\omega}$. Reason about engine torque, tire forces, breaking, etc. explicitly.
  - "Double-Track" Model: 4 tires, full 3D rigid body dynamics, suspension etc. Reason about body roll, weight transfer, agressive cornering, racing, drifting, off-road driving.
  - Tire models can go from simple (no-slip), to medium (Coulomb), to complex (contact patch, deformation, nonlinear friction).

* State-of-the-Art Nonlinear MPC
  - Dynamic bicycle model with good tire models gets you really far.
  - Online MPC with IPOPT at ~ 50 Hz

* The Frozen Robot Problem:

- We want the MPC controller to reason about coupled behavior with other drivers.
- Current systems make lots of assumptions. Simplest: other cars will continue at constant velocity over MPC horizon.
- Works well for highway driving
- Breaks down in scenarios with tighter coupling between cars (e.g. ramp merging).

* Game-Theoretic Trajectory Optimization

- High-level idea: Assume other cars are also solving an MPC problem like ours.
- Solve a joint optimization problem for all cars simultaneously
- One version of this idea: "Nash Equilibrium"

$$\bar{X} = \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^n \end{bmatrix}, \quad \bar{U} = \begin{bmatrix} u^1 \\ u^2 \\ \vdots \\ u^n \end{bmatrix}$$

← States + inputs for all cars stacked

$$\min_{\bar{X}, u^i} \; J^i(\bar{X}, u^i)$$

← cost for player i

$$\text{s.t.} \quad \begin{aligned} D(\bar{X}, \bar{U}) &= 0 \\ C(\bar{X}, \bar{U}) &\leq 0 \end{aligned}$$

} dynamics + collision constraints for all cars

- We get n (number of cars) of these problems that we must solve simultaneously

- Interpretation: No player can unilaterally improve their cost

- Good model of non-cooperative behavior

- Cost functions can capture driver behavior like aggressiveness etc.

* Solution Strategy (ALGAMES)

- Form Augmented Lagrangian for each player's 1st-order necessary conditions:

$$L^i(\bar{x}, \bar{u}, \lambda^i, \mu^i) = J^i(\bar{x}, u^i) + \frac{\Delta}{2}\|D(\bar{x}, \bar{u})\|_2^2 + \lambda^{iT} D(\bar{x}, \bar{u})$$

$$+ \frac{\rho}{2}\|C(\bar{x}, \bar{u})^+\|_2^2 + \mu^{iT} C(\bar{x}, \bar{u})$$

$$\Rightarrow \nabla_{u^i} L^i = 0$$

- Stack FON conditions for all players:

$$\begin{bmatrix} \nabla_{u^1} L^1 \\ \nabla_{u^2} L^2 \\ \vdots \\ \nabla_{u^n} L^n \end{bmatrix} = 0$$

- Solve with Newton's method

- With good implementation, can run at ~50 Hz for ~6 cars

- Generates human-like interaction strategies