Topics

- Classical optimal control
- Nonlinear stability
- Model predictive control
- Adaptive dynamic programming

First part of course

- Dynamic programming for Markov chains
- Classical Reinforcement learning
- Deeplearning / neural nets
- Deep reinforcement learning

Second part of course

- Special Topics

Homeworks Due Fridays

Office hours

- Wednesday 11 AM

Project

- Project proposal
- Project Update
- Presentation
- Report

Scribing:

Each student must "scribe" a lecture

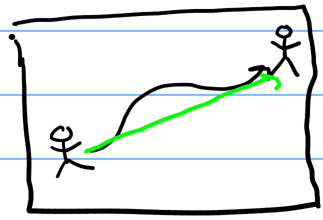
- Take good notes
- Type them into latex (I'll provide a basic template)

Dynamic models

- Integrator model

$$\dot{x}(t) = u(t) \Rightarrow x(t) = x(0) + \int_0^t u(\tau)\,d\tau$$

$$\dot{x}(t) = \frac{dx(t)}{dt}$$



← take the shortest path

u(t) are speed commands.

$x(t)$ ← state

$u(t)$ ← input / control / action

- Newton's Laws

$$x = \begin{bmatrix} p \\ v \end{bmatrix} \qquad \begin{aligned} \dot{p}(t) &= v(t) \\ \dot{v}(t) &= \frac{u(t)}{m} \end{aligned} \Rightarrow \dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u(t)$$

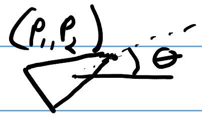
$a(t) = \dot{v}(t)$

$m a(t) = u(t)$

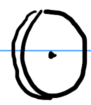
Standard form of Liner systems

$$\dot{x}(t) = Ax(t) + Bu(t)$$

- Unicycle vehicle model



reason for name



$$x(t) = \begin{bmatrix} p_1(t) \\ p_2(t) \\ \theta(t) \end{bmatrix}$$

$$u(t) = \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix}$$

$$\begin{bmatrix} \dot{p}_1(t) \\ \dot{p}_2(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} v(t)\cos\theta(t) \\ v(t)\sin\theta(t) \\ \omega(t) \end{bmatrix}$$

Move vehicle To (0,0,0) at T while minimizing $\int_0^T (v(t)^2 + \omega(t)^2)\,dt$ w/ $|v(t)| \le 1$, $|\omega(t)| \le 1$

general form nonlinear system

$$\frac{dx(t)}{dt} = \dot{x}(t) = f(x(t), u(t)) \Rightarrow x(t) - x(0) = \int_0^t f(x(\tau), u(\tau))\,d\tau$$

Euler Integration

$$\frac{dx(t)}{dt} = \lim_{h \to 0} \frac{x(t+h) - x(t)}{h} = f(x(t), u(t))$$

so for "small" h

$$\frac{x(t+h) - x(t)}{h} \approx f(x(t), u(t))$$

$$\Rightarrow \quad x(t+h) \approx x(t) + h f(x(t), u(t))$$

Euler integrator iterates this:

$$\underbrace{x((k+1)h)}_{\text{next state}} = \underbrace{x(kh)}_{\text{current state}} + h f(x(kh), \underbrace{u(kh)}_{\text{current input}})$$

What is optimal control?

Move in a way to minimize a cost

costs can encode

- Fuel use
- Path length
- Monetary loss
- Smoothness

Main idea is to encode desired behavior into cost and let algorithms handle the optimization

---

Linear Quadratic Regulator

$$\min \int_0^T \left(x(t)^T Q x(t) + u(t)^T R u(t) \right) dt + x(T)^T S_T x(T)$$

s.t. $\dot{x}(t) = Ax(t) + Bu(t)$

Require S_T, Q are positive semidefinite

R is positive definite

LQR example

Newton's Laws

$x = \begin{bmatrix} p \\ v \end{bmatrix}$

want
- $p(T) \approx 0$
- $v(t)$ never very big
- $u(t) \leftarrow$ Force, also never very big

$$x(T)^T S_T^T x(T) = \alpha \, p(T)^2 = \begin{bmatrix} p(T) \\ v(T) \end{bmatrix}^T \begin{bmatrix} \alpha & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p(T) \\ v(T) \end{bmatrix}$$

$\uparrow$ Tunable coefficient $\hookrightarrow S_T^T$

$x^T Q x = \beta v^2$ achived by setting $Q = \begin{bmatrix} 0 & 0 \\ 0 & \beta \end{bmatrix}$

$u^T R u = u^2$ given by $R = 1$