Lecture 11: Control III

Professor Katie Driggs-Campbell

February 22, 2024

ECE484: Principles of Safe Autonomy

Administrivia

- Team formation due this week (check campuswire)
- Upcoming due dates:
  - HW1 and MP1 due Friday 2/23
  - HW2 and MP2 due Friday 3/01
  - Project Pitches in class 3/05 and 3/07
- Exam has been moved to 4/18 at 7pm in 1013 and 1015 ECEB

Project Expectations

- GEM Track or F1tenth Track
  - Given RGB-D input, detect the track and follow the path
  - Add at least one additional feature!
- GRAIC Track
  - Given ground truth reference waypoints and obstacle positions, complete circuits around the tracks
  - Add one additional cool feature!

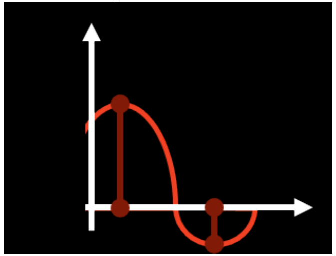
- **Now: Finalize Teams and Tracks**
- **Now: Start Safety Training for Hardware Teams**
- 03/05: Project Pitch Presentation
- 03/29 or 04/05: Mid Check-in
- 04/23: Final Presentation
- 05/03: Video Due

Today's Plan (Part 1)

- Take a look at PID controllers
- Build up waypoint following using the models discussed previously
- Introduce some advanced control techniques

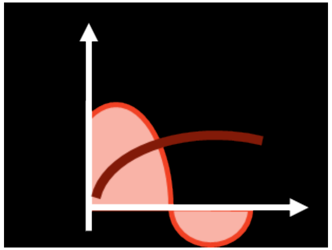
PID Controllers

- Proportional



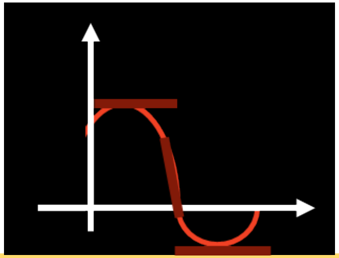
$$u = k_p e$$

- Integral

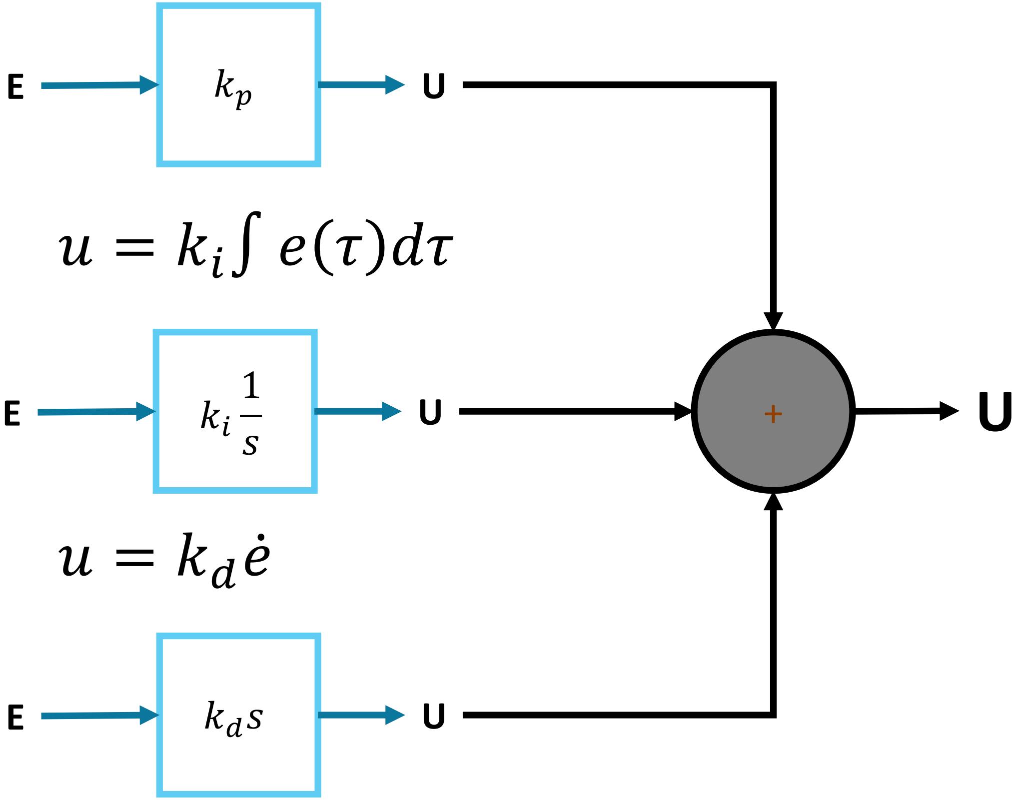


$$u = k_i \int e(\tau)d\tau$$

- Derivative



$$u = k_d \dot{e}$$



Linear Error Dynamics

$$a_p e^{(p)} + \cdots + a_2 \ddot{e} + a_1 \dot{e} + a_0 e = 0$$

$$x_p = e^{(p)}, \quad x_3 = \ddot{e}, \quad x_2 = \dot{e}, \quad x_1 = e$$

$$\dot{x}_p = -\frac{a_0}{a_p} x_1 - \frac{a_1}{a_p} x_2 - \cdots$$

$$\dot{x} = Ax$$

$$\begin{bmatrix} \dot{x}_1 \\ \vdots \\ \dot{x}_p \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ -a_0/a_p & & \cdots & & -a_{p-1}/a_p \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_p \end{bmatrix}$$

Viewing as a Second Order System

- The second order system is: $\ddot{e} + c_1\dot{e} + c_2 e = 0$
- In standard form, we write:

$$\ddot{e}(t) + 2\xi\omega_n\dot{e}(t) + \omega_n^2 e(t) = 0$$

where ξ is the *damping ratio* and ω_n is the *natural frequency*

- The eigenvalues are given as:

$$\lambda_{1,2} = -\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$$

- Note that the system is stable iff ω_n and ξ are positive

Second Order Dynamics: Cases

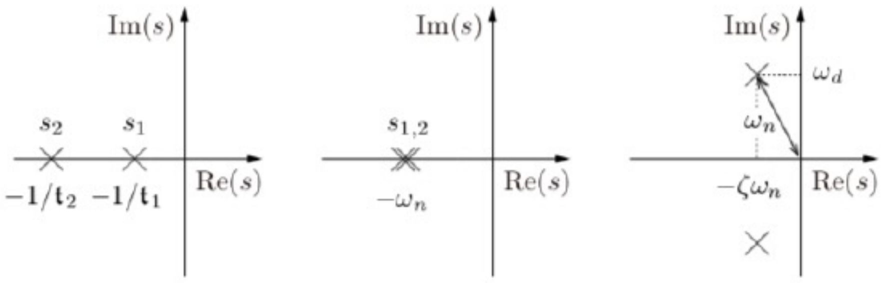
Recall:

$\lambda_{1,2} = -\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$

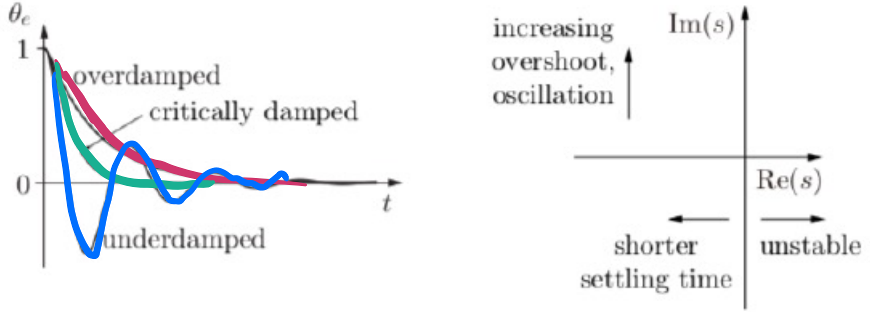
- Overdamped: $\zeta > 1$
  - Roots s_1 and s_2 are distinct
  - $x(t) = c_1 e^{s_1 t} + c_2 e^{s_2 t}$
  - Time constant is the less negative root
- Critically damped: $\zeta = 1$
  - Roots s_1 and s_2 are equal and real
  - $x(t) = (c_1 + c_2 t)e^{-\omega_n t}$
  - Time constant is given by $1/\omega_n$
- Underdamped: $\zeta < 1$
  - Roots are complex conjugates:

    $$s_{1,2} = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$$

  - $x(t) = (c_1 \cos\omega_d t + c_2 \sin\omega_d t)e^{-\zeta\omega_n t}$



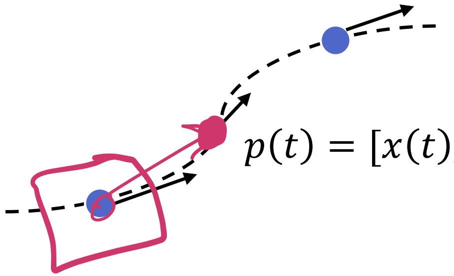
overdamped ($\zeta > 1$) critically damped ($\zeta = 1$) underdamped ($\zeta < 1$)



On to PID for path following

Path following control

- The path followed by a robot can be represented by a *trajectory or path* parameterized by time
  - → from a higher-level planner, map, or perception system
- Defines the desired instantaneous pose $p(t)$



$$p(t) = [x(t), y(t), \theta(t)]$$

Open-loop waypoint following

- We can write an open-loop controller for a robot that is naturally controlled via angular velocity, such as a differential-drive robot:

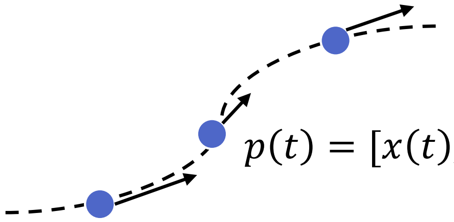
$$u_{\omega,OL}(t) = \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} = \begin{bmatrix} \sqrt{\dot{x}(t)^2 + \dot{y}(t)^2} \\ \dot{\theta}(t) \end{bmatrix}$$

- We can write an open-loop controller for a robot with car-like steering:

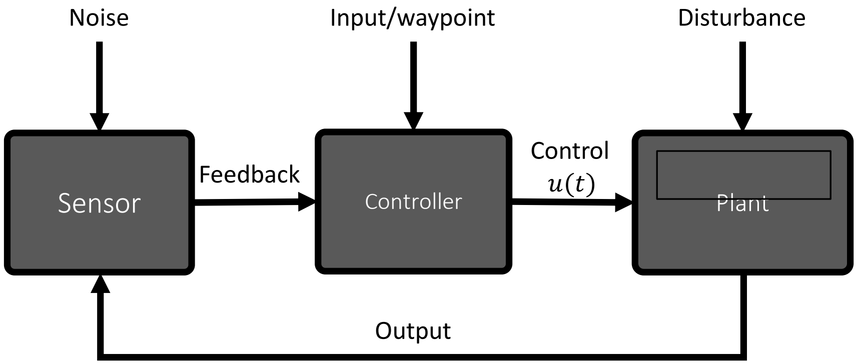
$$u_{\kappa,OL}(t) = \begin{bmatrix} v(t) \\ \kappa(t) \end{bmatrix} = \begin{bmatrix} \sqrt{\dot{x}(t)^2 + \dot{y}(t)^2} \\ \dfrac{\dot{\theta}(t)}{\sqrt{\dot{x}(t)^2 + \dot{y}(t)^2}} \end{bmatrix}$$

Path following control

- The path followed by a robot can be represented by a *trajectory or path* parameterized by time
  - → from a higher-level planner, map, or perception system
- Defines the desired instantaneous pose $p(t)$

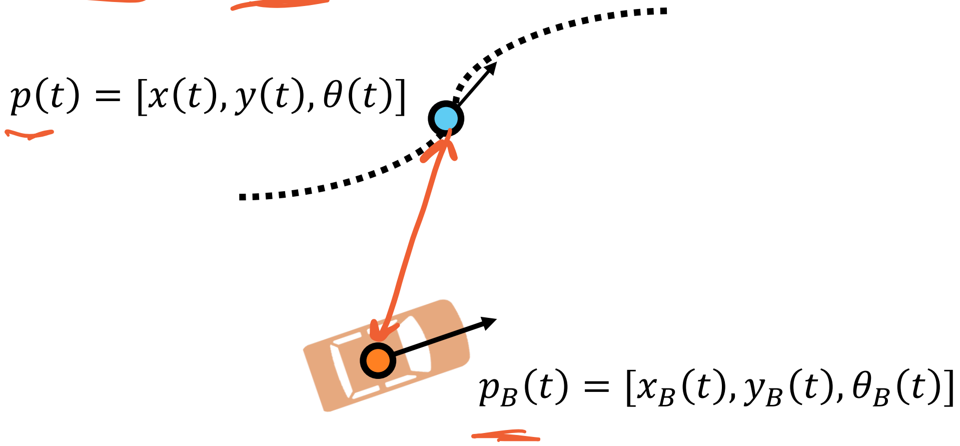


$$p(t) = [x(t), y(t), \theta(t)]$$



Path following control

- Desired instantaneous pose $p(t)$
- How to define error between actual pose $p_B(t)$ and desired pose $p(t)$ in the form of $y_d(t) - y(t)$?



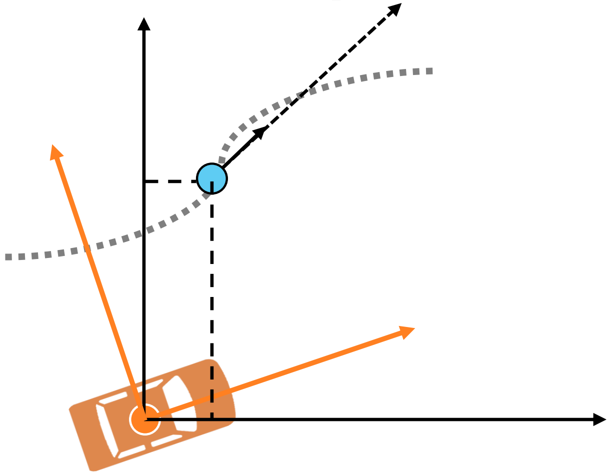
Path following control

The error vector measured vehicle coordinates

$$e(t) = [\delta_s(t), \delta_n(t), \delta_\theta(t), \delta_v(t)]$$

$[\delta_s, \delta_n]$ define the coordinate errors in the vehicle's reference frame: along track error and cross track error

$p(t) = [x(t), y(t), \theta(t), v(t)]$



$p_B(t) = [x_B(t), y_B(t), \theta_B(t), v_B(t)]$

Path following control

The error vector measured vehicle coordinates

$$\mathrm{e}(t) = [\delta_s(t), \delta_n(t), \delta_\theta(t), \delta_v(t)]$$

$[\delta_s, \delta_n]$ define the coordinate errors in the vehicle's reference frame: along track error and cross track error

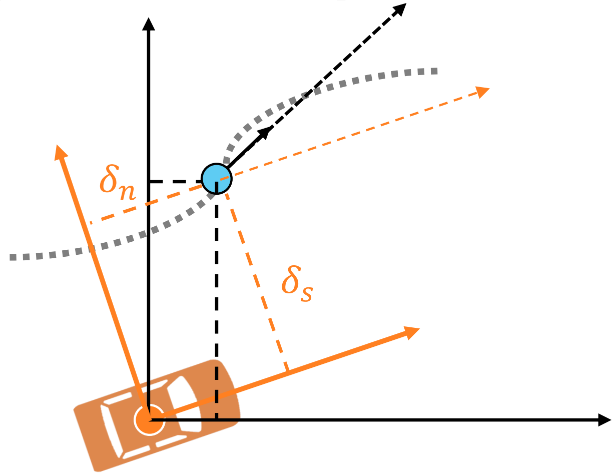
- Along track error: distance ahead or behind the target in the instantaneous direction of motion.

$$\delta_s = \cos(\theta_B(t))\,(x(t) - x_B(t)) + sin(\theta_B(t))\,(y(t) - y_B(t))$$

- Cross track error: portion of the position error orthogonal to the intended direction of motion

$$\delta_n = -\sin(\theta_B(t))\,(x(t) - x_B(t)) + cos(\theta_B(t))\,(y(t) - y_B(t))$$

$$p(t) = [x(t), y(t), \theta(t), v(t)]$$



$$p_B(t) = [x_B(t), y_B(t), \theta_B(t), v_B(t)]$$

Path following control

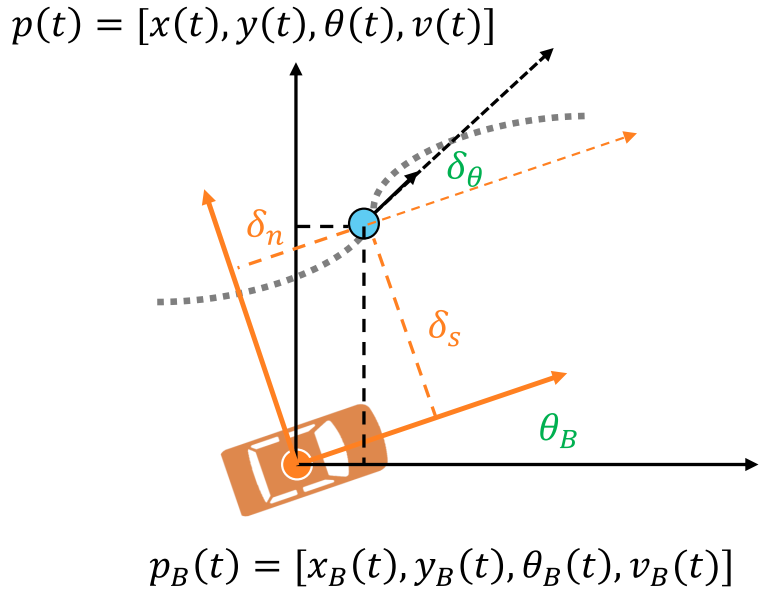
The error vector measured vehicle coordinates

$$e(t) = [\delta_s(t), \delta_n(t), \delta_\theta(t), \delta_v(t)]$$

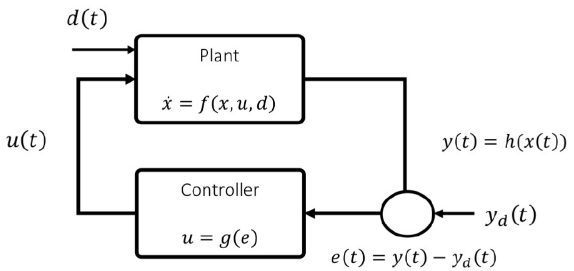
$[\delta_s, \delta_n]$ define the coordinate errors in the vehicle's reference frame: along track error and cross track error

- Along track error: distance ahead or behind the target in the instantaneous direction of motion.
  $$\delta_s = \cos(\theta_B(t))\,(x(t) - x_B(t)) + sin(\theta_B(t))\,(y(t) - y_B(t))$$
- Cross track error: portion of the position error orthogonal to the intended direction of motion
  $$\delta_n = -\sin(\theta_B(t))\,(x(t) - x_B(t)) + cos(\theta_B(t))\,(y(t) - y_B(t))$$
- Heading error: difference between desired and actual orientation and direction
  $$\delta_\theta = \theta(t) - \theta_B(t)$$
  $$\delta_v = v(t) - v_B(t)$$

→ Each of these errors match the form $y_d(t) - y(t)$



A simple P-controller example



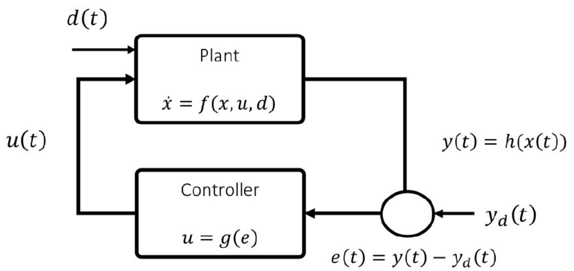
disturbance

- Given a simple system: $\dot{y}(t) = u(t) + d(t)$
- Using proportional (P) controller:

$$u(t) = -K_P e(t) = -K_P(y(t) - y_d(t))$$

$$\dot{y}(t) = -K_P y(t) + K_P y_d(t) + d(t)$$

A simple P-controller example



- Given a simple system: $\dot{y}(t) = u(t) + d(t)$
- Using proportional (P) controller:

$$u(t) = -K_P e(t) = -K_P(y(t) - y_d(t))$$

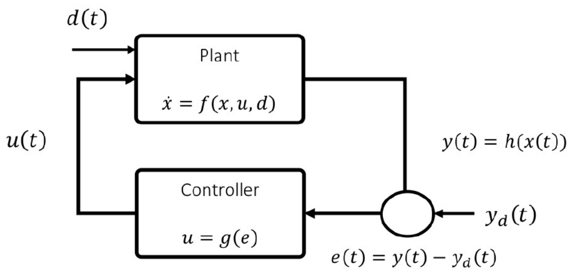
$$\dot{y}(t) = -K_P y(t) + K_P y_d(t) + d(t)$$

- Consider constant setpoint y_0 and disturbance d_{ss}

$$\dot{y}(t) = -K_P y(t) + K_P y_0 + d_{ss}$$

- What is the steady state output?

A simple P-controller example



- Given a simple system: $\dot{y}(t) = u(t) + d(t)$
- Using proportional (P) controller:

$$u(t) = -K_P e(t) = -K_P(y(t) - y_d(t))$$

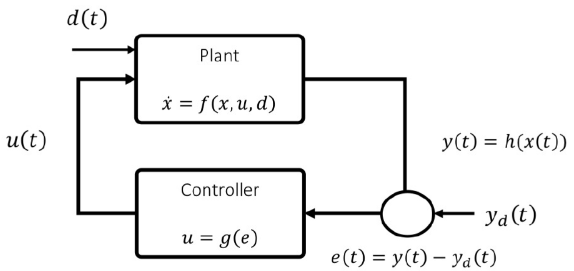
$$\dot{y}(t) = -K_P y(t) + K_P y_d(t) + d(t)$$

- Consider constant setpoint y_0 and disturbance d_{ss}

$$\dot{y}(t) = -K_P y(t) + K_P y_0 + d_{ss}$$

- What is the steady state output?
  - Set: $-K_P y(t) + K_P y_0 + d_{ss} = 0$
  - Solve for y_{ss}: $y(t) = \frac{d_{ss}}{K_P} + y_0$

A simple P-controller example

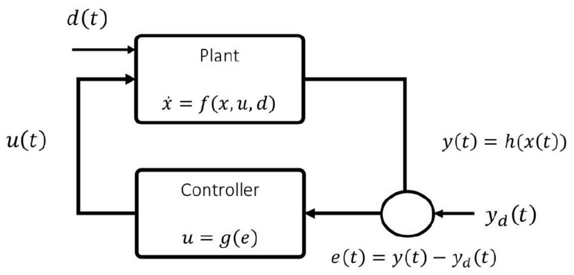


- Given a simple system: $\dot{y}(t) = u(t) + d(t)$
- Consider constant setpoint y_0 and disturbance d_{ss}

$$\dot{y}(t) = -K_P y(t) + K_P y_0 + d_{ss}$$

- Steady state output $y_{ss} = \frac{d_{ss}}{K_P} + y_0$

A simple P-controller example



- Given a simple system: $\dot{y}(t) = u(t) + d(t)$
- Consider constant setpoint y_0 and disturbance d_{ss}

$$\dot{y}(t) = -K_P y(t) + K_P y_0 + d_{ss}$$

- Steady state output $y_{ss} = \frac{d_{ss}}{K_P} + y_0$
- Transient behavior:

$$y(t) = y_0 e^{-t/T} + y_{ss}\left(1 - e^{-t/T}\right), T = 1/K_P$$

- To make steady state error small, we can increase K_P at the expense of longer transients

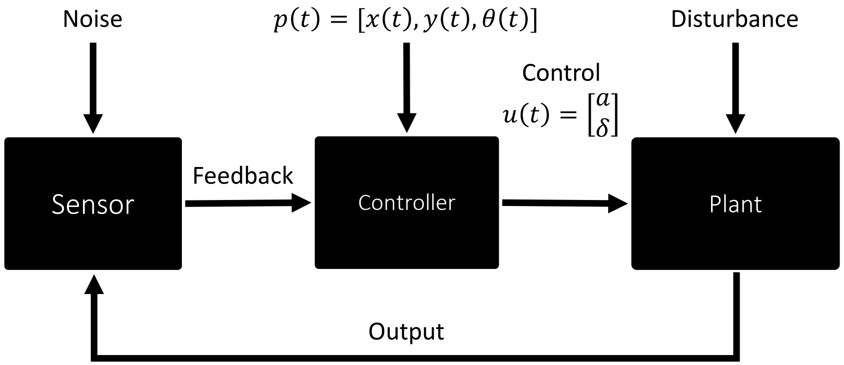
Control Law

Control input is given by $u = [a, \delta]^\top$

where a is the acceleration and δ is the steering angle

$$u = K \begin{bmatrix} \delta_s \\ \delta_n \\ \delta_\theta \\ \delta_v \end{bmatrix}$$

$$K = \begin{bmatrix} K_s & 0 & 0 & K_v \\ 0 & K_n & K_\theta & 0 \end{bmatrix}$$

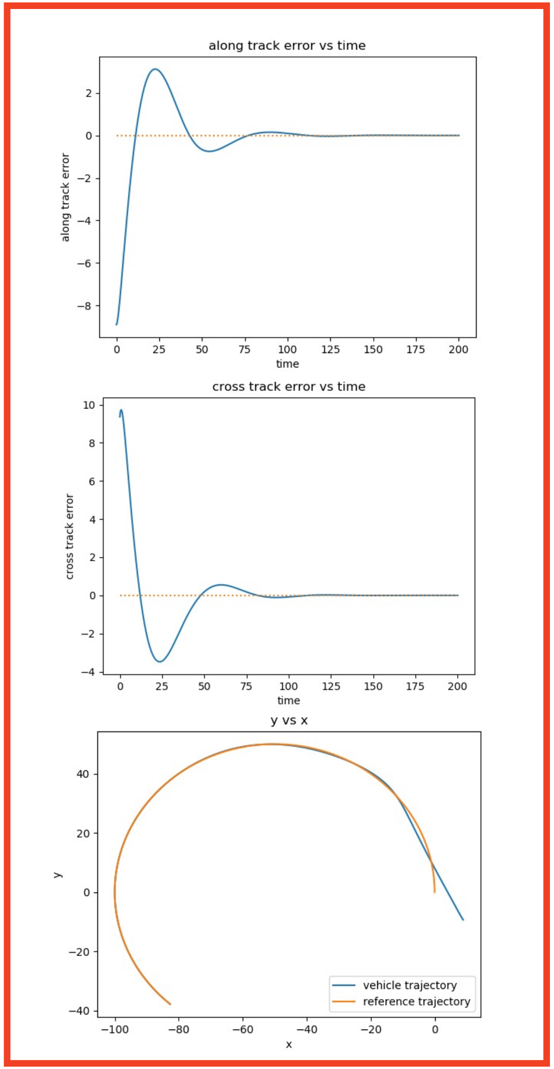


Control Law

$$K = \begin{bmatrix} K_s & 0 & 0 & K_v \\ 0 & K_n & K_\theta & 0 \end{bmatrix}$$

The pure-pursuit controller produced by this gain matrix performs a PD-control. It uses a PD-controller to correct along-track error.

The control on curvature is also a PD-controller for cross-track error because δ_θ is related to the derivative of δ_n.



Midpoint Summary

- Reviewed linear systems and stability of differential equations
- Looked at PID controllers as a way to regulate systems using state feedback
- Derived a waypoint following error dynamics
  - → This will be needed for MP2!

Advanced Control Topics

Today's Plan

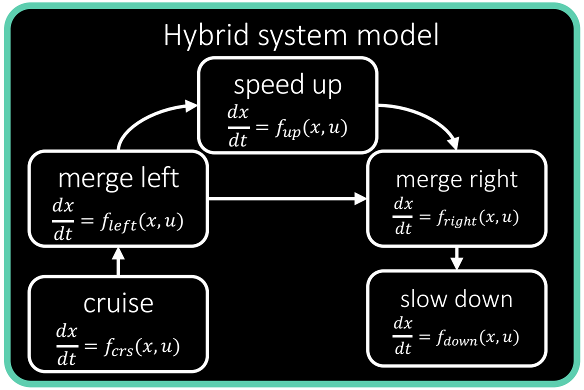
- Quick discussion of future topics in advanced control theory
- Introduction to optimal control
  - Linear Quadratic Regulation (LQR)
  - Model Predictive Control (MPC)
- End-to-end learning

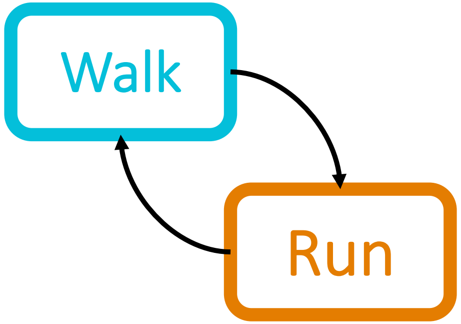
Today's Plan

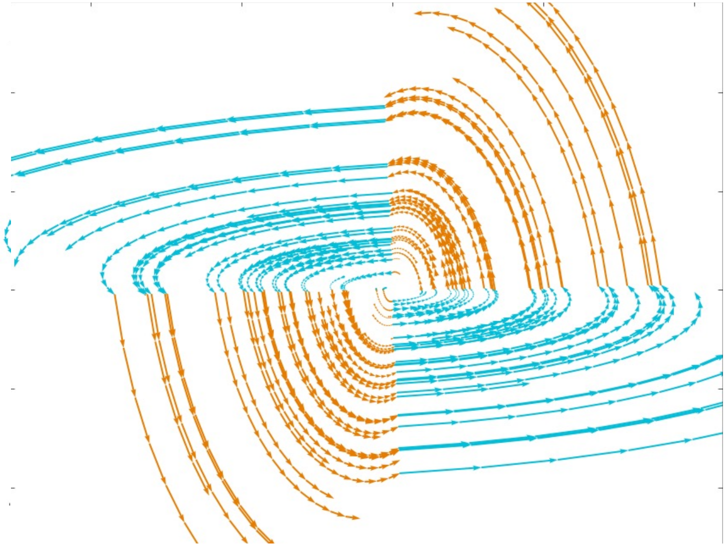
- Quick discussion of future topics in advanced control theory
- Introduction to optimal control
  - Linear Quadratic Regulation (LQR)
  - Model Predictive Control (MPC)
- End-to-end learning

Extensions from Control Theory

1. Hybrid Control
   - Given discrete modes of continuous behavior, can we guarantee stability?



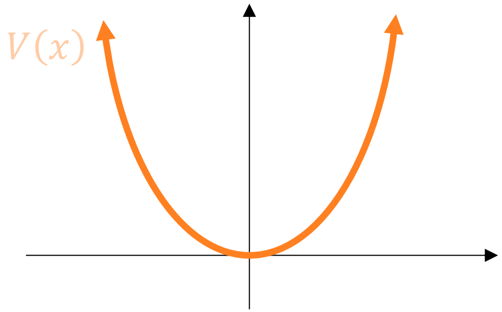




UIUC has one of the best
control programs in the country!
Consider some grad courses in this area!

Extensions from Control Theory

1. Hybrid Control
   - Given discrete modes of continuous behavior, can we guarantee stability?
2. Lyapunov Stability
   - The system is said to be Lyapunov stable about an equilibrium if
     $\forall \varepsilon > 0\ \exists \delta_\varepsilon > 0$ such that $|x_0| \leq \delta_\varepsilon \Rightarrow \forall t \geq 0, |\xi(x_0, t)| \leq \varepsilon$



Today's Plan

- Quick discussion of future topics in advanced control theory
- Introduction to optimal control
  - Linear Quadratic Regulation (LQR)
  - Model Predictive Control (MPC)
- End-to-end learning

Convex Optimization

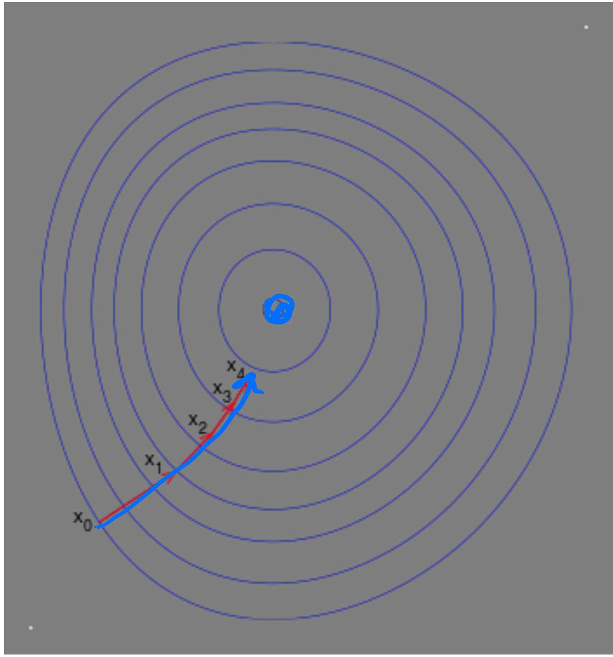
$$\underset{x}{\text{minimize}} \quad J(x)$$

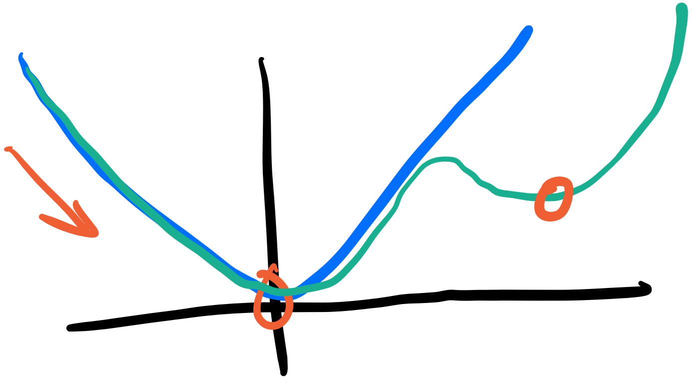
$$\text{subject to} \quad g_i(x) \le 0 \quad \forall i$$

$$h_j(x) = 0 \quad \forall j$$

if ↑ is convex,

→ every local min is a global min





Linear Quadratic Regulation (LQR)

$$x_{t+1} = Ax_t + Bu_t$$

$$J(x, u) = x_T^T Q_F x_T + \sum_t x_t^T Q x_t + u_t^T R u_t$$

where $Q, R, Q_F \in \mathbb{R}^{n \times n}$

Is Optimal Enough?

Deploying a PID Controller

Video Credit: Jonathan Hui

Is Optimal Enough?

Deploying a PID Controller

Model Predictive Control

Model Predictive Control

Receding Horizon Approach:

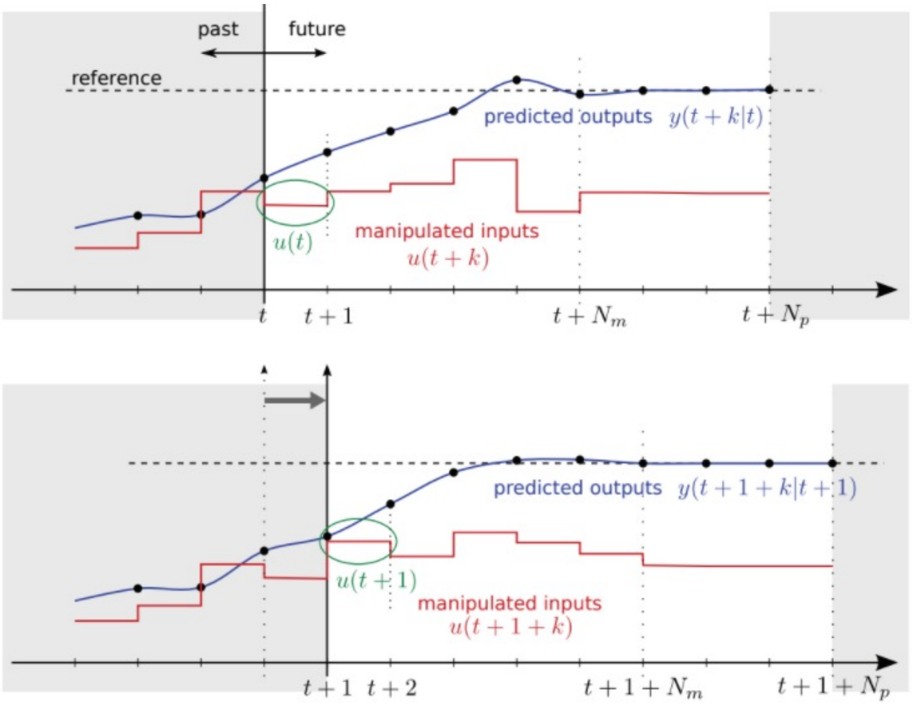
$$\text{minimize}_{x,u}\ J(x,u)$$

$$\text{subject to}\quad x_t = f(x_{t-1}, u_{t-1})$$

$$x_0 = x_{init}, x_T = x_G$$

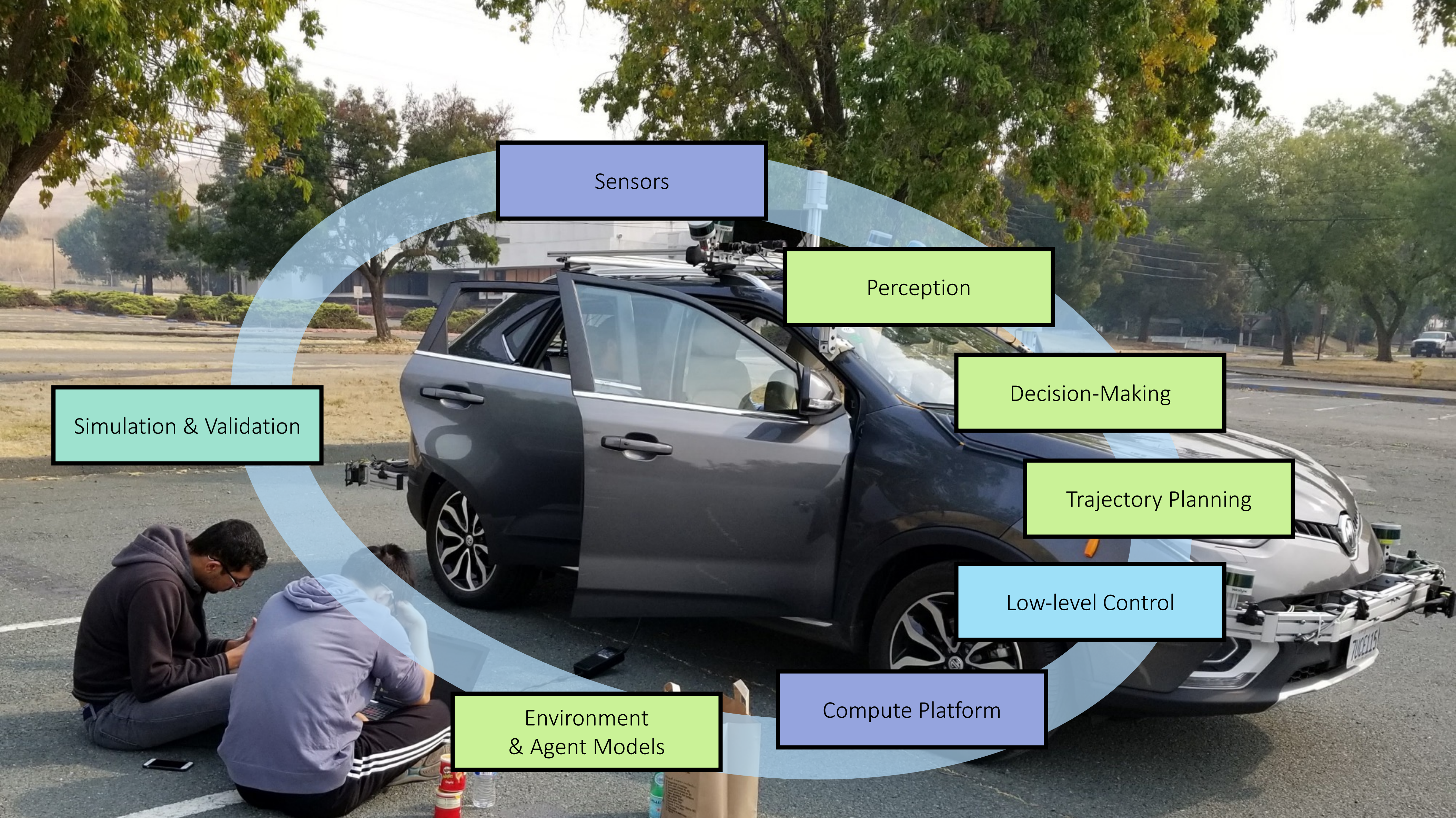
$$\underline{u} \leq u \leq \bar{u}$$

$$\underline{x} \leq x \leq \bar{x}$$



Optimize over time horizon T, execute u_1, optimize again with updated information.

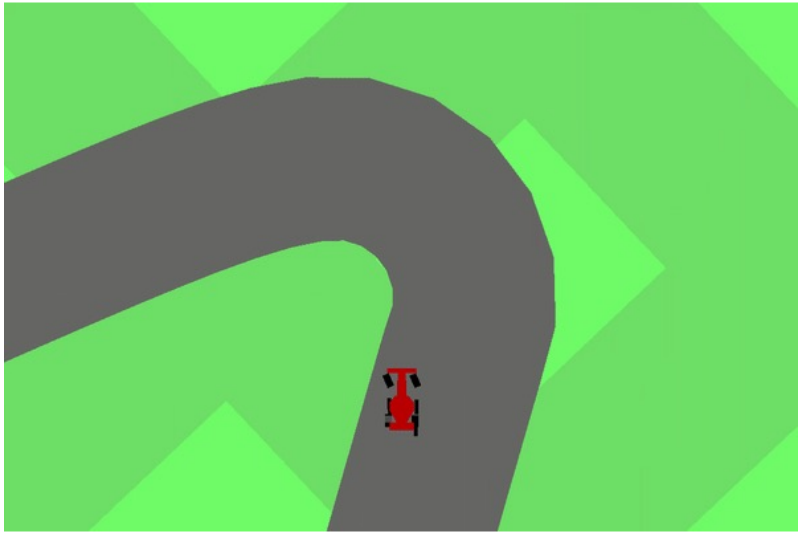
Image Credit: F. Borrelli

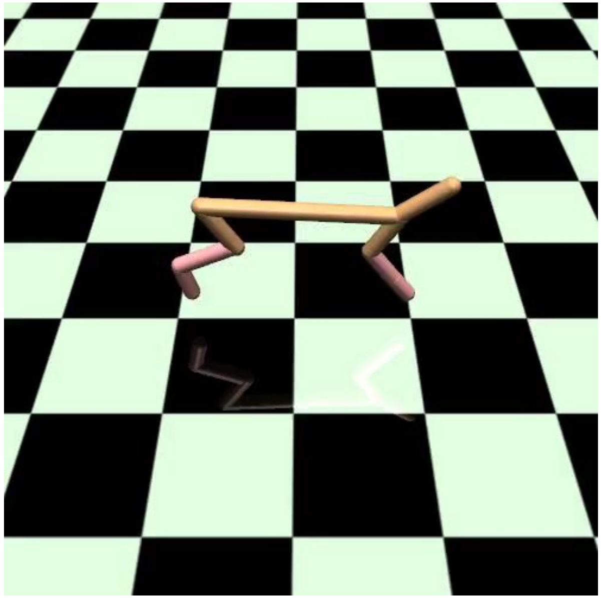

Sensors
Perception
Decision-Making
Simulation & Validation
Trajectory Planning
Low-level Control
Environment
& Agent Models
Compute Platform

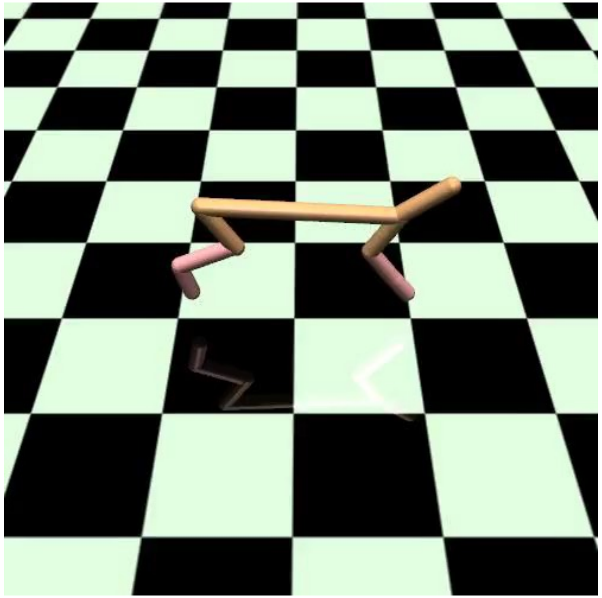
Today's Plan

- Quick discussion of future topics in advanced control theory
- Introduction to optimal control
  - Linear Quadratic Regulation (LQR)
  - Model Predictive Control (MPC)
- End-to-end learning

RL Approaches: Hand Specifying Rewards







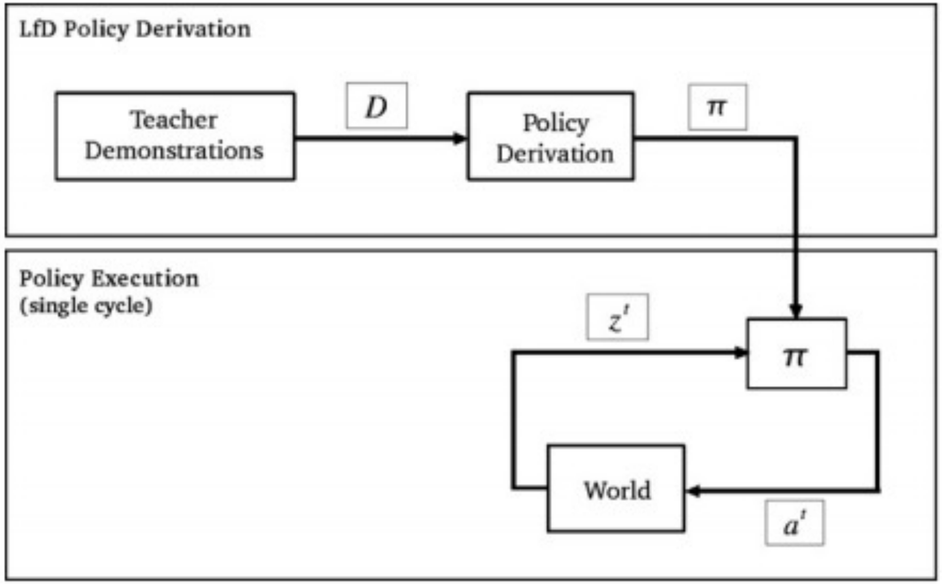
Video Credit: A. Irpan and https://notanymike.github.io/Solving-CarRacing/

Experience vs. Demonstrations

Reinforcement Learning

Demonstrations (sort of)

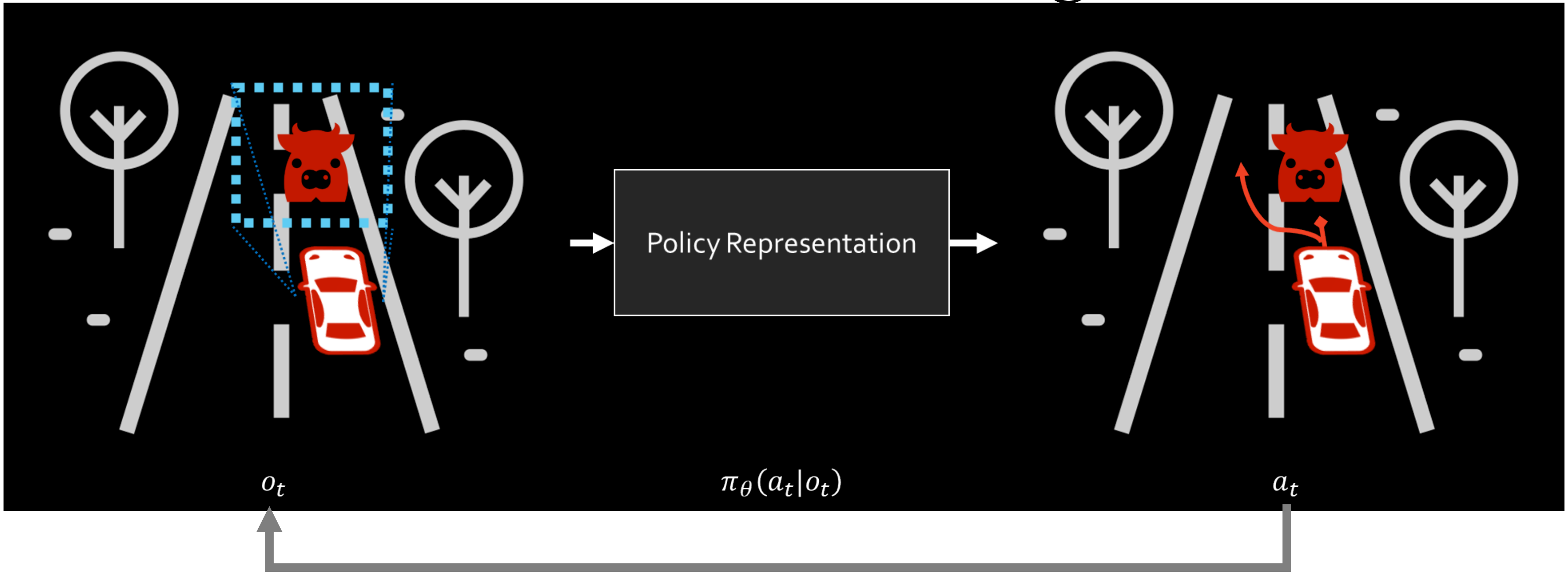
LfD: Framework and Design Choices

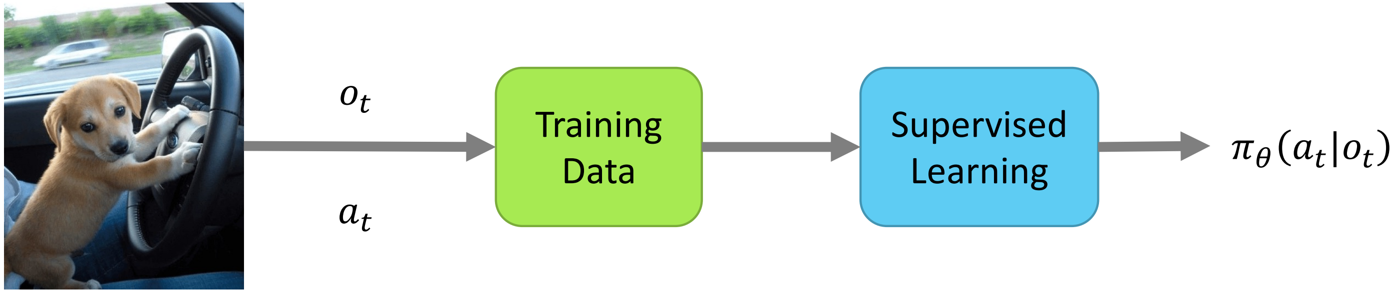


- Demonstration approach
  - Choice of demonstrator (expert)
  - Demonstration technique (offline, online, iterative)
- Problem space continuity
- Dataset gathering (and limitations)
  - Correspondence (recording, embodiment)
  - Demonstration (teleoperation, shadowing)
- Policy derivation

B. Argall, et al., "A survey of robot learning from demonstration," Robotics and Autonomous Systems, 2009.

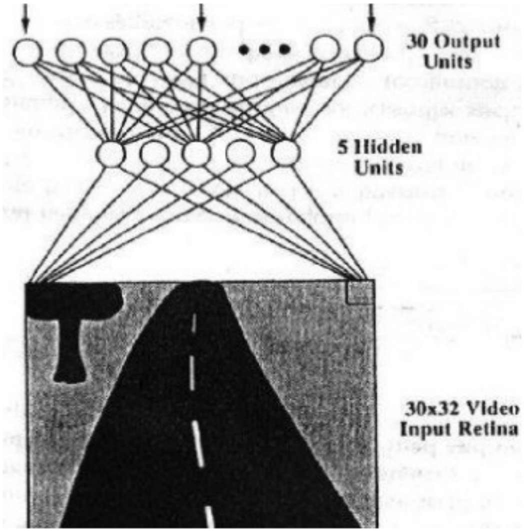
Behavior Cloning

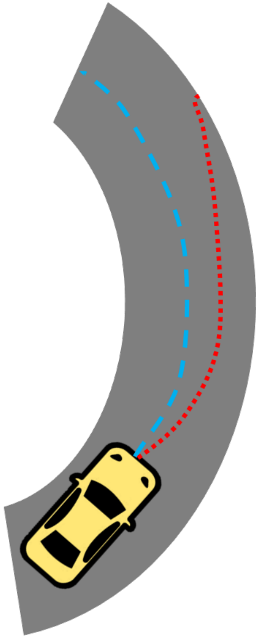




Behavior Cloning

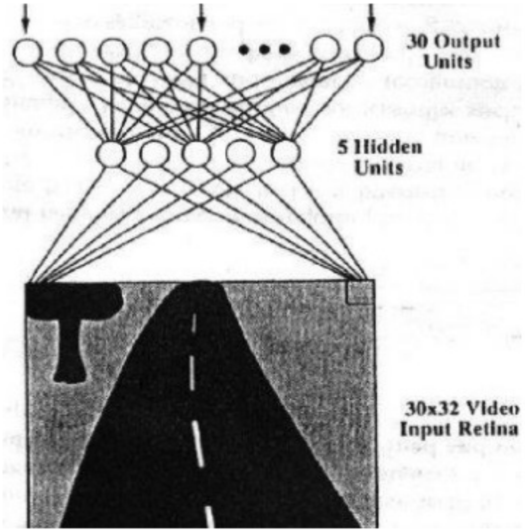
ALVINN: Autonomous Land Vehicle In a Neural Network (1989)



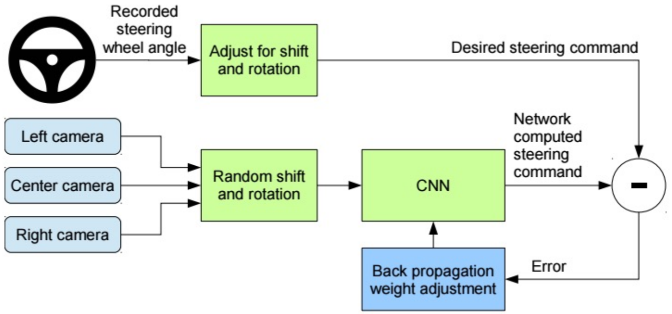


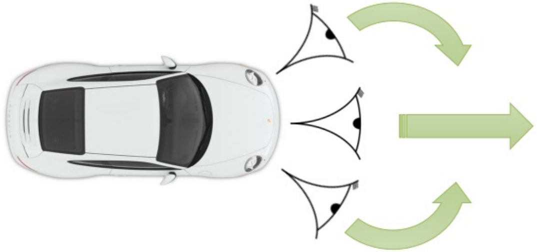
Behavior Cloning

ALVINN: Autonomous Land Vehicle In a Neural Network (1989)



End-to-End Deep Learning for Self-Driving Cars (2016)





Pomerleau, Dean A. "Alvinn: An autonomous land vehicle in a neural network." *Advances in neural information processing systems.* 1989.
M. Bojarski, et al. "End to end learning for self-driving cars." arXiv:1604.07316 (2016).

NN Policy Baseline

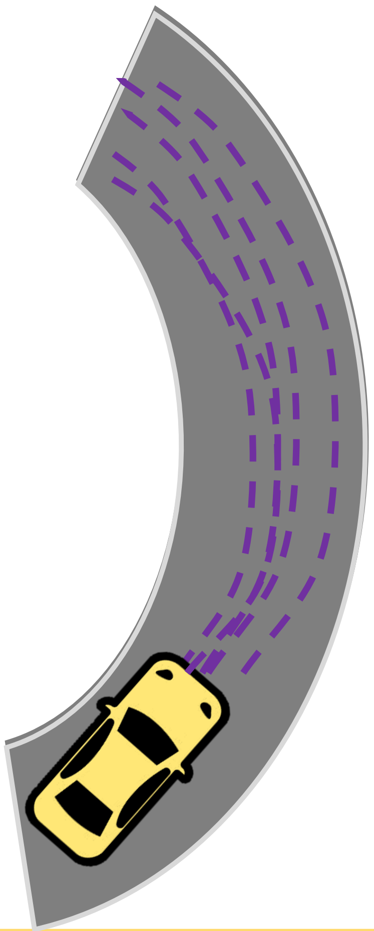
HG-DAgger:

Interactive Imitation Learning with Human Experts

M. Kelly, C. Sidrane, K. Driggs-Campbell, M. Kochenderfer

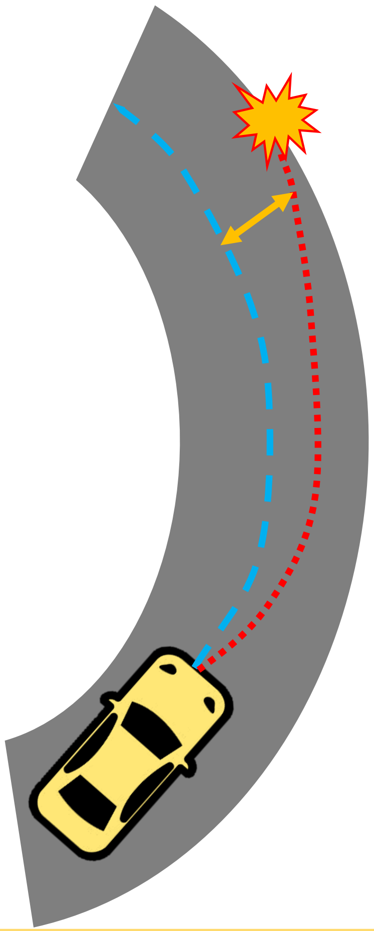
(Deep) Imitation Learning

- Given sample trajectories from an expert, try to learn the underlying policy



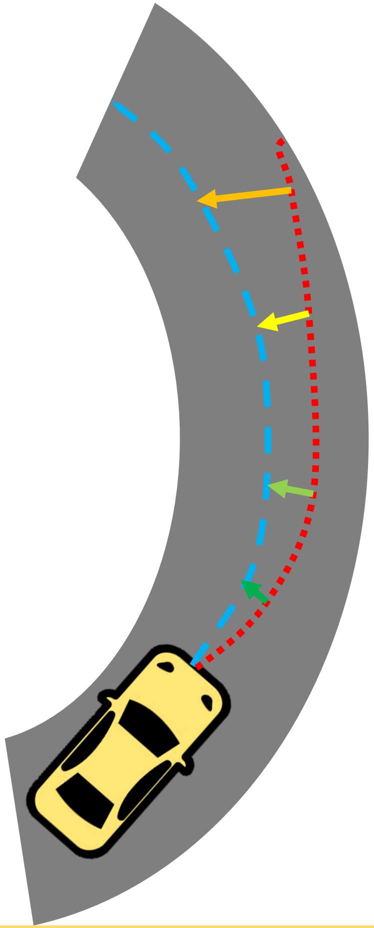
(Deep) Imitation Learning

- Given sample trajectories from an expert, try to learn the underlying policy
- Tends to suffer from distribution shift, compounding errors, model mismatch

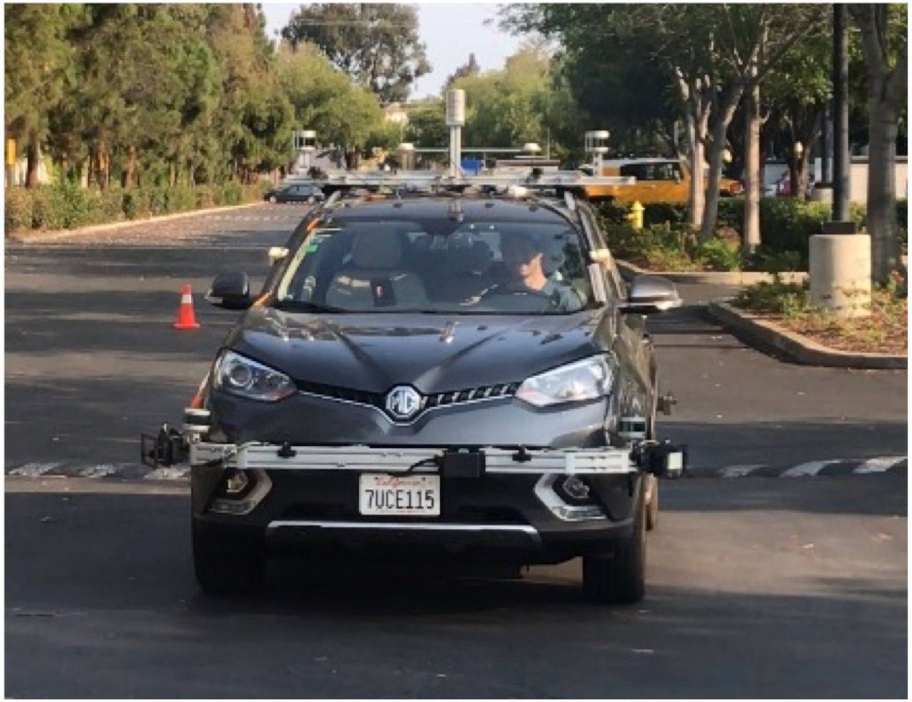


(Deep) Imitation Learning

- Given sample trajectories from an expert, try to learn the underlying policy
- Tends to suffer from distribution shift, compounding errors, model mismatch
- By improving how we collect the data, we can improve the resulting policy!



Human-Gated Imitation Learning



HG-DAgger:
Interactive Imitation Learning with Human Experts

M. Kelly, C. Sidrane, K. Driggs-Campbell, M. Kochenderfer

Summary

- Introduced a few advanced topics on model-based control
- Discussed learning and end-to-end (model-free) approaches
- Note that all of the methods discussed require some low-level controller (i.e., PID) and some high-level input (i.e., decision-making)
- Did not discuss the safety implications of different control methods! What do you think are the hazards and advantages of different approaches?
- *Next time:* Filtering and localization!

Extra Slides

Inverse Reinforcement Learning

- Given an optimal trajectory, we want to find the cost function:

$$\xi_D \rightarrow \mathcal{U}: \Xi \rightarrow \mathbb{R}_+ \ s.t. \mathcal{U}[\xi_D] \leq \mathcal{U}[\xi], \forall \xi$$

- Rewrite as: $\mathcal{U}[\xi_D] \leq \min_{\xi} \mathcal{U}[\xi]$ → Suffers from trivial solutions!
- Modify to find cost function that gives minimum cost by a margin:

$$\mathcal{U}[\xi_D] \leq \min_{\xi} \mathcal{U}[\xi] - l(\xi, \xi_D), \text{ where } l(\xi, \xi_D) = \begin{cases} 0 \text{ if } \xi = \xi_D \\ 1 \text{ otherwise} \end{cases}$$

- To make this hold true for the *maximum margin*:

$$\max_{\mathcal{U}} \min_{\xi} \mathcal{U}[\xi] - l(\xi, \xi_D) - \mathcal{U}[\xi_D]$$

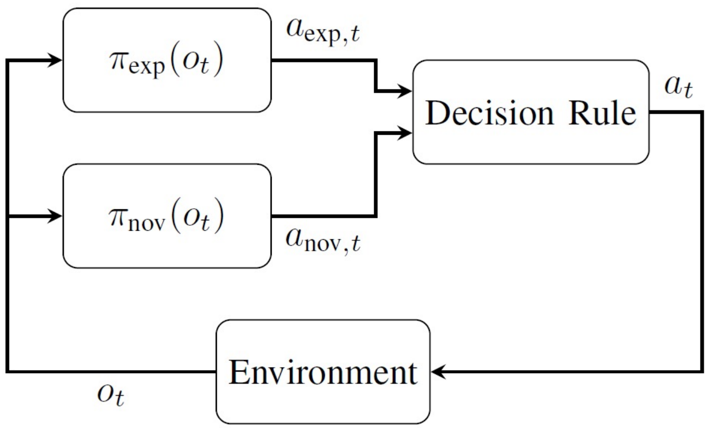
$$\min_{\mathcal{U}} \left[\mathcal{U}[\xi_D] - \min_{\xi} [\mathcal{U}[\xi] - l(\xi, \xi_D)] + \lambda R(\mathcal{U}) \right]$$

- To solve this problem, parameterize the function $\mathcal{U}$ → often a linear combination of features

P. Abbeel and A. Ng. "Apprenticeship learning via inverse reinforcement learning." International Conference on Machine learning. 2004.

DAgger: Dataset Aggregation

1. Train π_{nov} from human data $\mathcal{D}$
2. Run π_{nov} to get dataset $\mathcal{D}_{\pi_{nov}}$
3. Obtain corrected labels
4. Aggregate: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_{\pi_{nov}}$
5. Repeat!

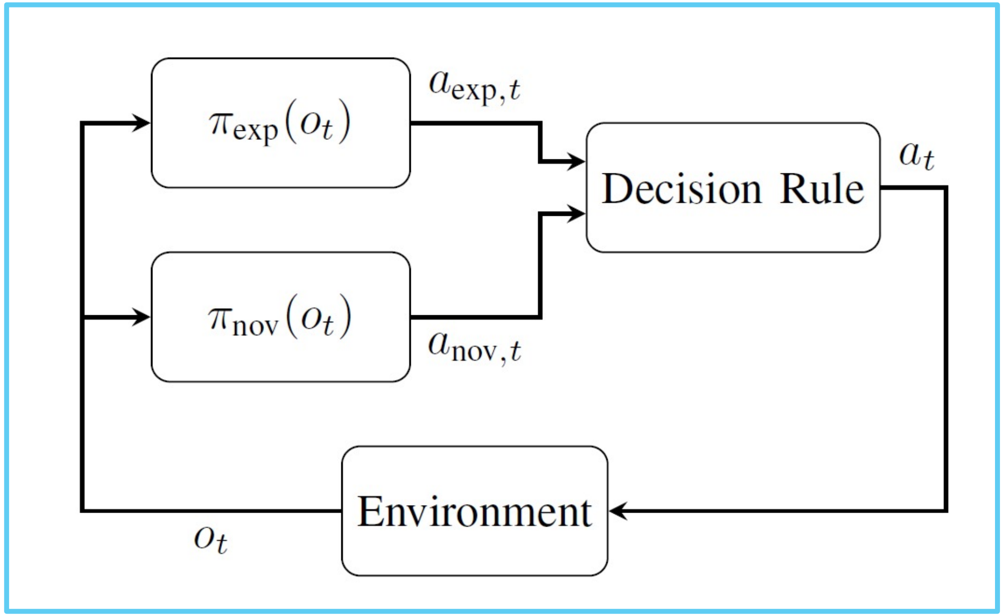


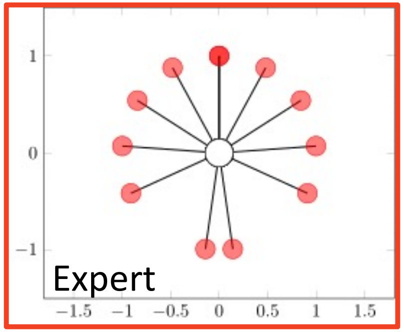
Algorithm 2 VanillaDAgger Decision Rule

procedure DR(o_t, i, β_0, λ)
  $a_{\text{nov},t} \leftarrow \pi_{\text{nov}}(o_t)$
  $a_{\text{exp},t} \leftarrow \pi_{\text{exp}}(o_t)$
  $\beta_i \leftarrow \lambda^i \beta_0$
  $z \sim \text{Uniform}(0, 1)$
  if $z \leq \beta_i$
    return $a_{\text{exp},t}$
  else
    return $a_{\text{nov},t}$

S. Ross, G. Gordon, and A. Bagnell. "A reduction of imitation learning and structured prediction to no-regret online learning." International Conference on Artificial Intelligence and Statistics. 2011.

"Safe" Imitation Learning



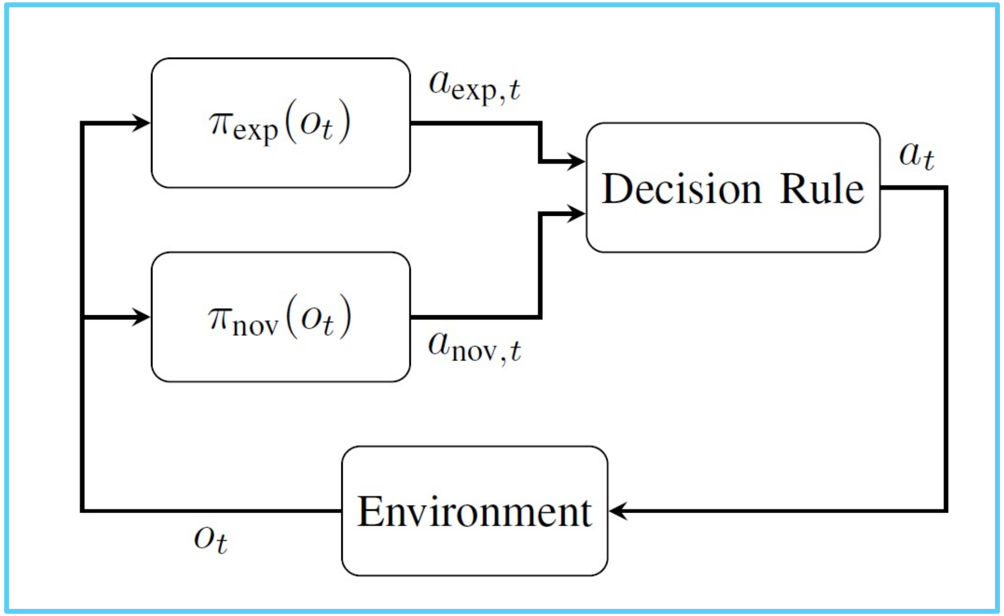


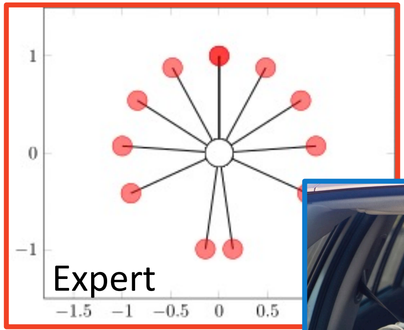
Online Training

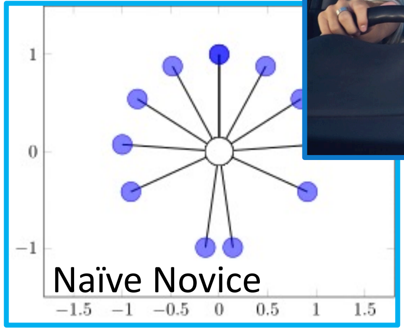
Naïve Novice

Trained Novice

“Safe” Imitation Learning

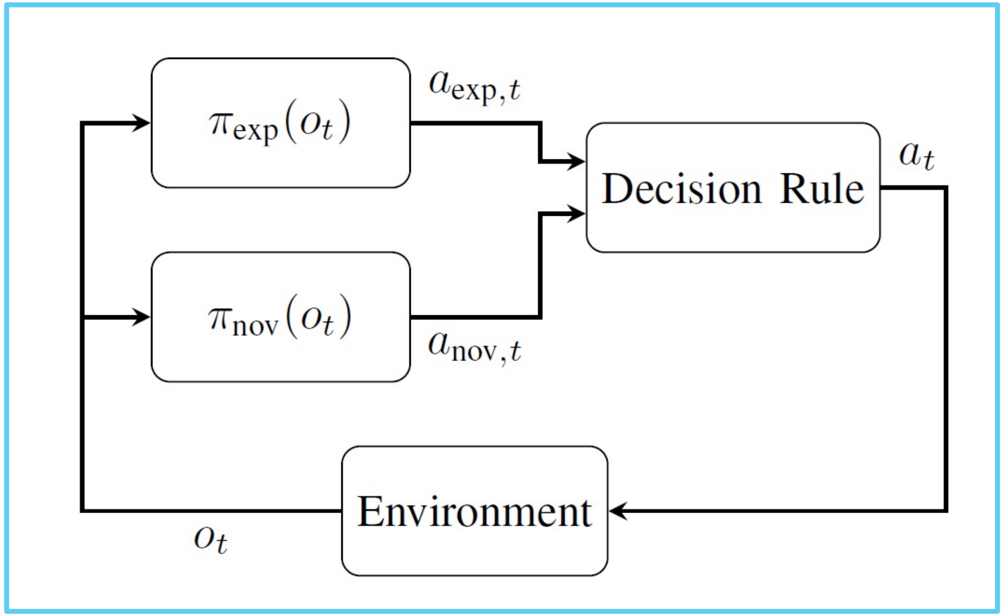


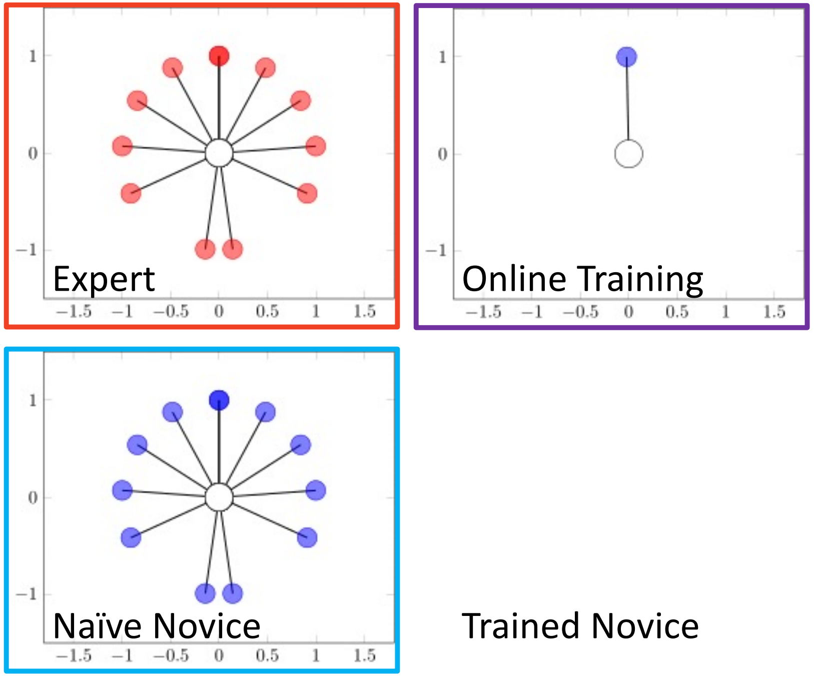




Trained Novice

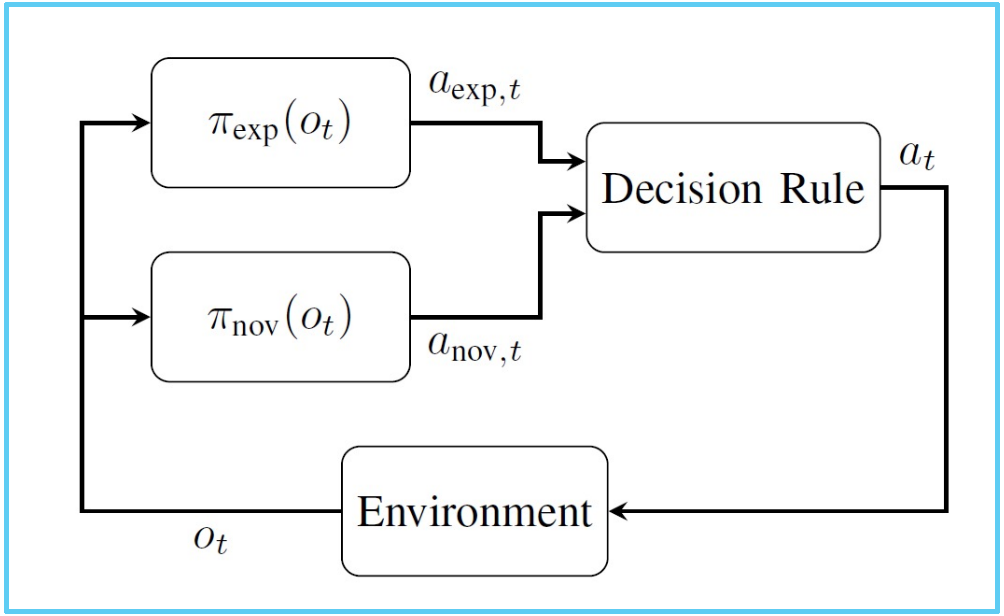
"Safe" Imitation Learning

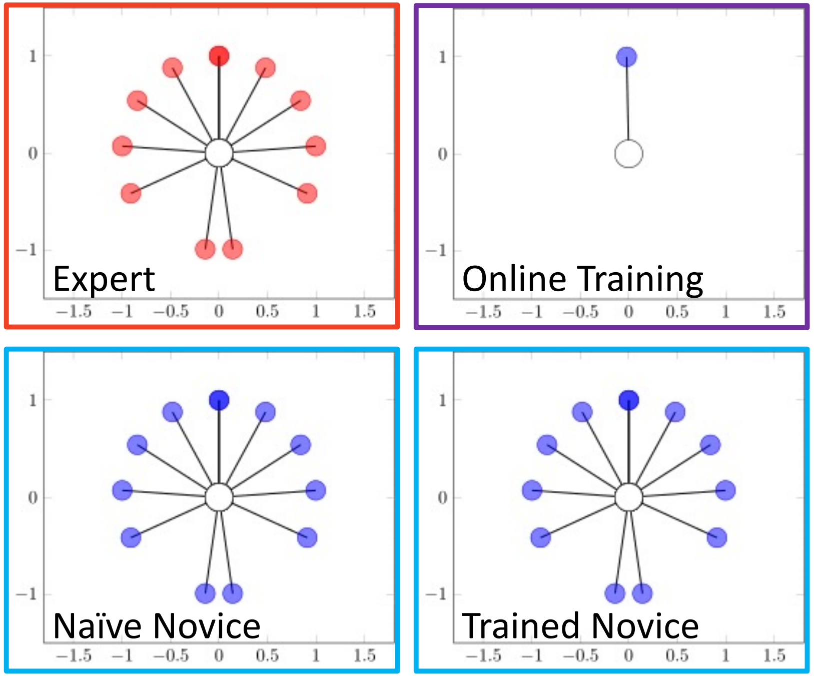




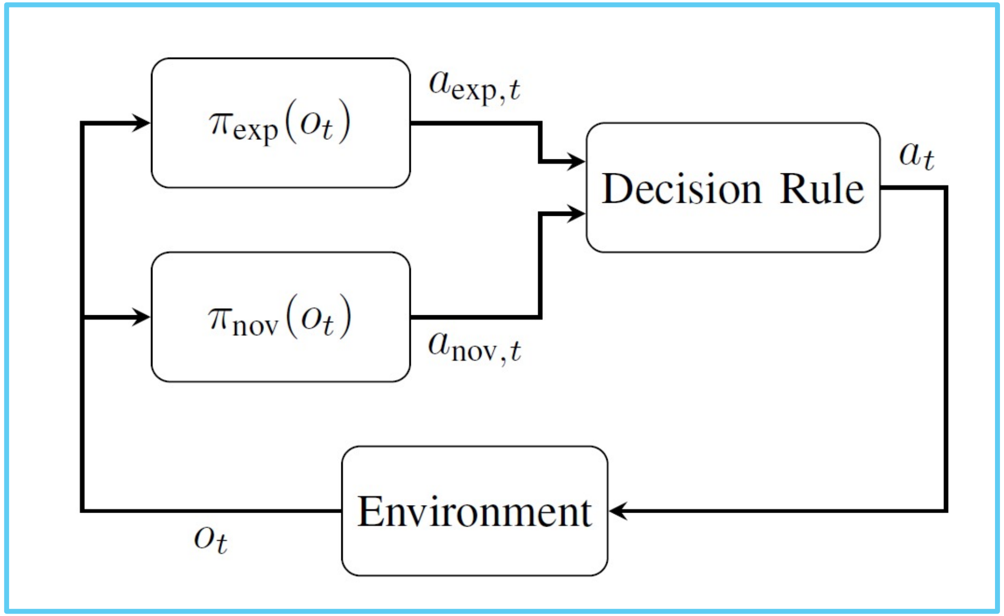
Visualization Credit: K. Menda

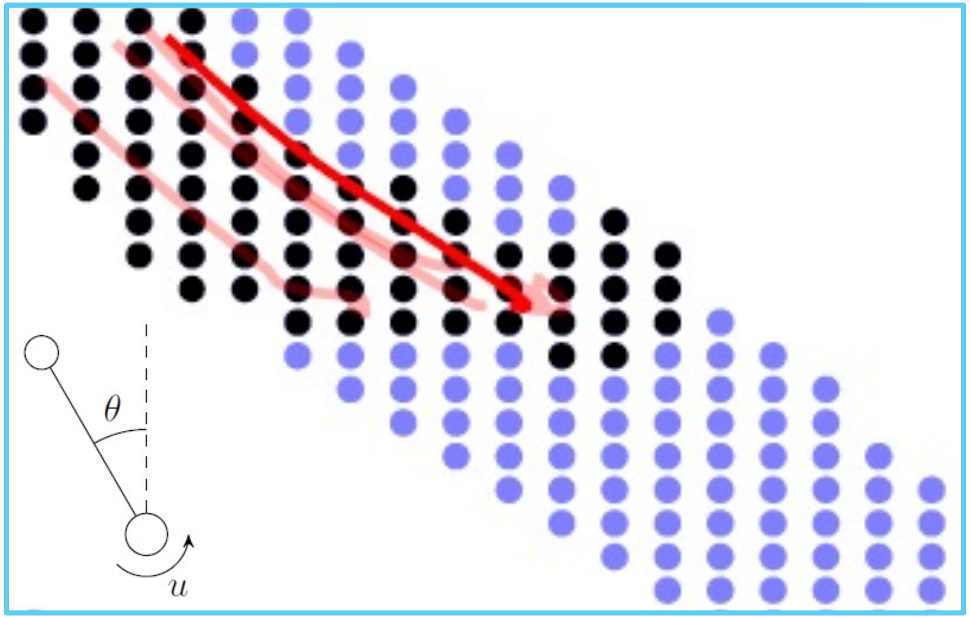
"Safe" Imitation Learning





Safe Imitation Learning





Visualization Credit: K. Menda

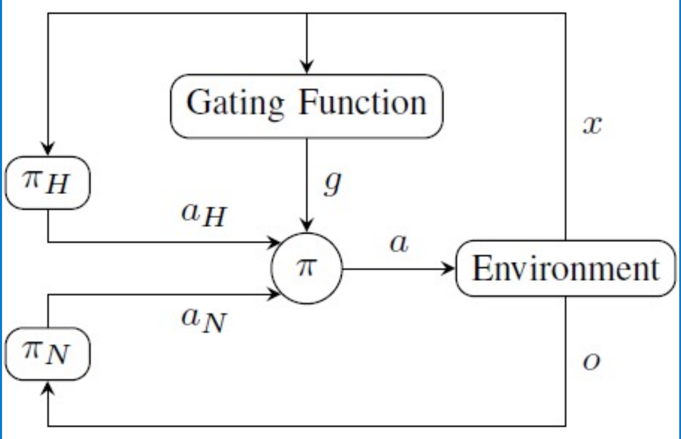
Methods for Determining the Decision Rule?

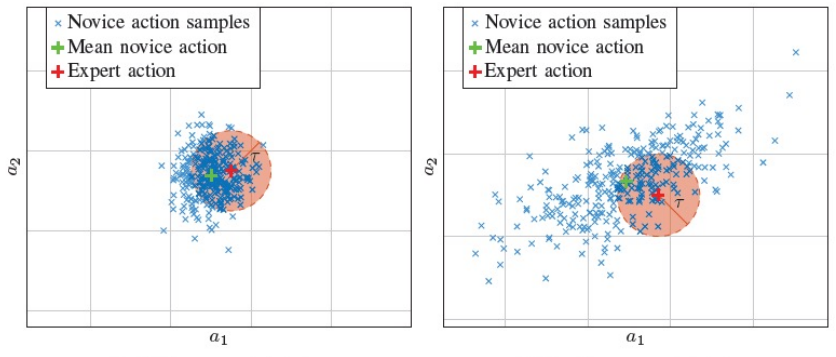
Algorithm 3 SAFEDAGGER* Decision Rule

1: **procedure** DR(o_t, τ)
2: $\quad a_{\mathrm{nov},t} \leftarrow \pi_{\mathrm{nov}}(o_t)$
3: $\quad a_{\mathrm{exp},t} \leftarrow \pi_{\mathrm{exp}}(o_t)$
4: $\quad$ **if** $\|a_{\mathrm{nov},t} - a_{\mathrm{exp},t}\|^2 \leq \tau$
5: $\qquad$ **return** $a_{\mathrm{nov},t}$
6: $\quad$ **else**
7: $\qquad$ **return** $a_{\mathrm{exp},t}$

Algorithm 4 EnsembleDAgger Decision Rule

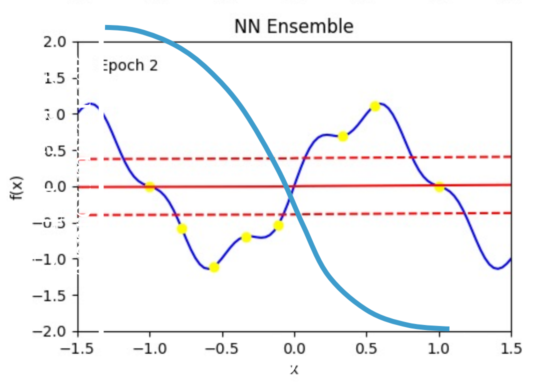
1: **procedure** DR(o_t, τ, χ)
2: $\quad \bar{a}_{\mathrm{nov},t}, \sigma^2_{a_{\mathrm{nov},t}} \leftarrow \pi_{\mathrm{nov}}(o_t)$
3: $\quad a_{\mathrm{exp},t} \leftarrow \pi_{\mathrm{exp}}(o_t)$
4: $\quad \hat{\tau} \leftarrow \|\bar{a}_{\mathrm{nov},t} - a_{\mathrm{exp},t}\|^2$
5: $\quad \hat{\chi} \leftarrow \sigma^2_{a_{\mathrm{nov},t}}$
6: $\quad$ **if** $\hat{\tau} \leq \tau$ **and** $\hat{\chi} \leq \chi$
7: $\qquad$ **return** $\bar{a}_{\mathrm{nov},t}$
8: $\quad$ **else**
9: $\qquad$ **return** $a_{\mathrm{exp},t}$





(a) Well-represented state

(b) Poorly-represented state

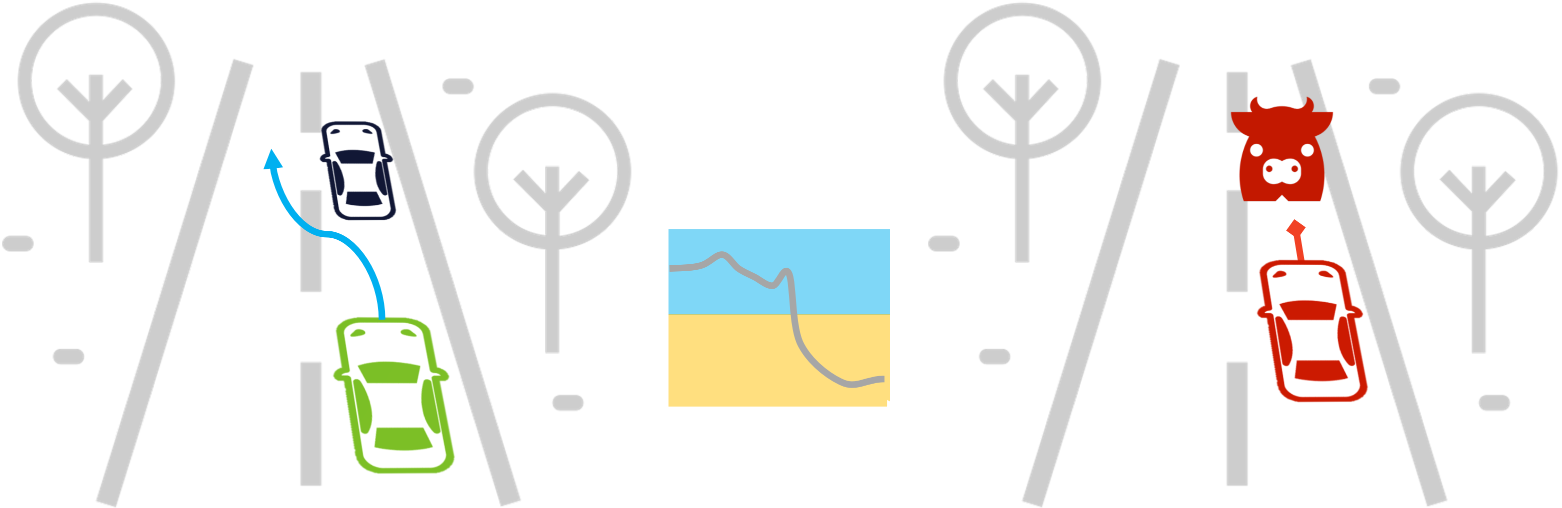


hang, a K. Cho. "Query-efficient imitation learning for end-to-end autonomous driving." arXiv:1605.06450, 2016.
Menda,

Self-Driving Demonstration

High Confidence in NN Policy

Unseen scenario → Resume control



HG-DAgger:
Interactive Imitation Learning with Human Experts

M. Kelly, C. Sidrane, K. Driggs-Campbell, M. Kochenderfer