Lecture 9: PID Control

Professor Katie Driggs-Campbell

March 2, 2021

ECE484: Principles of Safe Autonomy

Administrivia

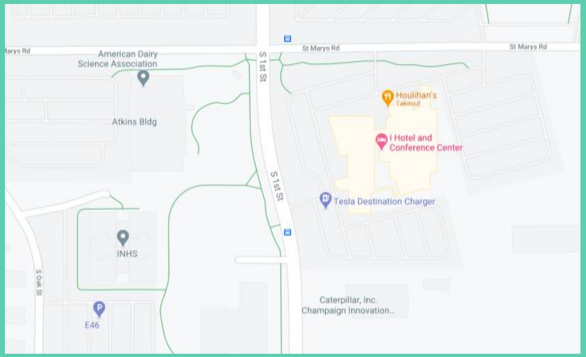
- Safety training information posted on discord
- MP1 due this week
  - Demo due Thursday
  - Report due Friday
- Milestone report due Friday 3/19 by 5pm
  - Rubric posted tonight

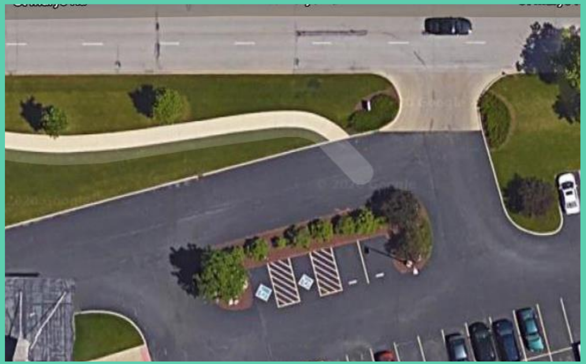
Today’s Plan

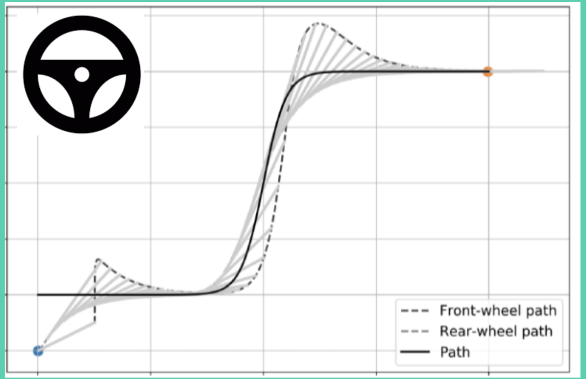
- Review some properties of dynamical systems and differential equations
- Take a look at PID controllers
- Build up to waypoint following using the models discussed last week!

Typical planning and control modules

- Global navigation and planner
  - Find paths from source to destination with static obstacles
  - Algorithms: Graph search, Dijkstra, Sampling-based planning
  - Time scale: Minutes
  - Look ahead: Destination
  - Output: reference center line, semantic commands
- Local planner
  - Dynamically feasible trajectory generation
  - Dynamic planning w.r.t. obstacles
  - Time scales: 10 Hz
  - Look ahead: Seconds
  - Output: Waypoints, high-level actions, directions / velocities
- Controller
  - Waypoint follower using steering, throttle
  - Algorithms: PID control, MPC, Lyapunov-based controller
  - Lateral/longitudinal control
  - Time scale: 100 Hz
  - Look ahead: current state
  - Output: low-level control actions







Dynamical Systems Model

Describe behavior in terms of instantaneous laws:

$$\frac{dx(t)}{dt} = \dot{x}(t) = f\big(x(t), u(t)\big)$$

// $x[t+1] = f(x[t], u[t])$

where $t \in \mathbb{R}$, $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, and $f\colon \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ gives the dynamics / transition function

$$\dot{x} = Ax$$

Examples

$\dot{x} = Ax + Bu \qquad u = -Kx$

$= Ax + BKx$

$= (A - BK)x$

$$\begin{bmatrix}\dot{x_2}\\ \dot{x_1}\end{bmatrix} = \begin{bmatrix}-1/4 & -2/5\\ 3 & -1/4\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

$\lambda_1 = -0.25 - i1.10$
$\lambda_2 = -0.25 + i1.10$

$$\begin{bmatrix}\dot{x_2}\\ \dot{x_1}\end{bmatrix} = \begin{bmatrix}1/4 & -2/5\\ 3 & -1/4\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

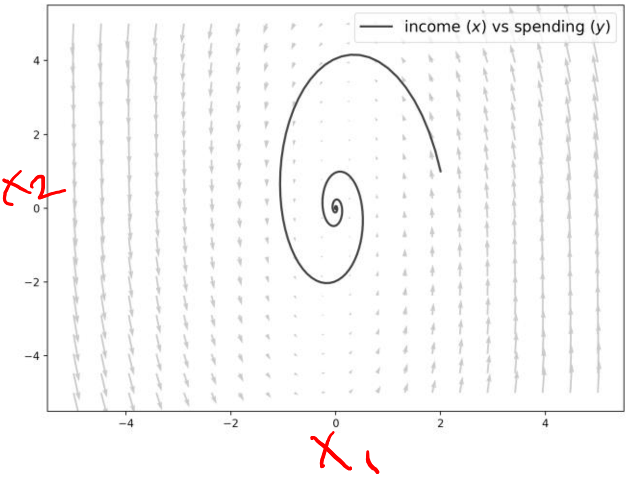
$\lambda_1 = +i0.1066$
$\lambda_2 = -i0.1066$

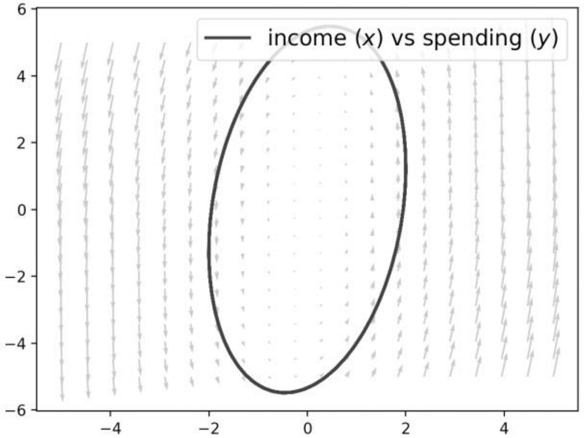
$$\begin{bmatrix}\dot{x_2}\\ \dot{x_1}\end{bmatrix} = \begin{bmatrix}1/2 & -2/5\\ 3 & -1/4\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

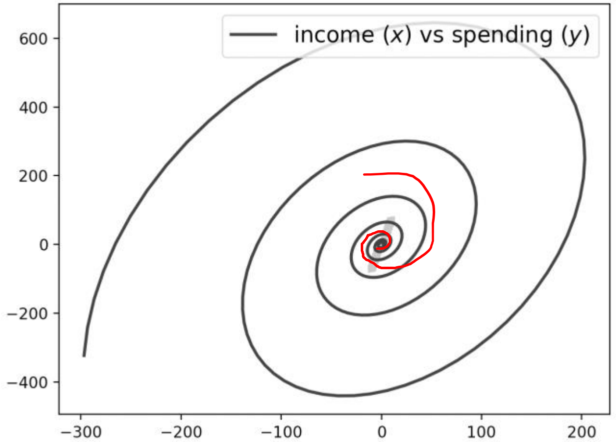
$\lambda_1 = 0.125 + i1.029$
$\lambda_2 = -0.125 - i1.029$

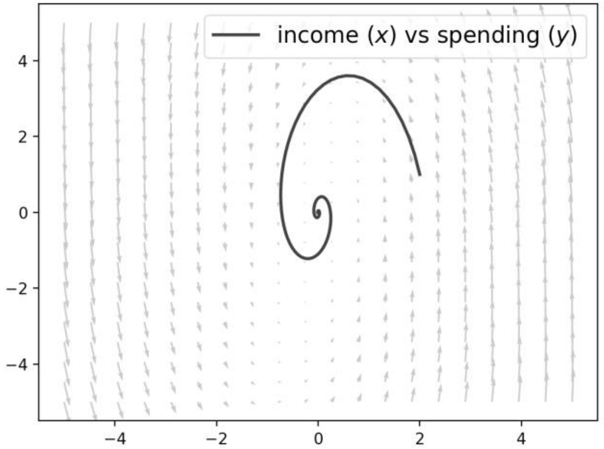
$$\begin{bmatrix}\dot{x_2}\\ \dot{x_1}\end{bmatrix} = \begin{bmatrix}-1/4 & -2/5\\ 3 & -1/2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

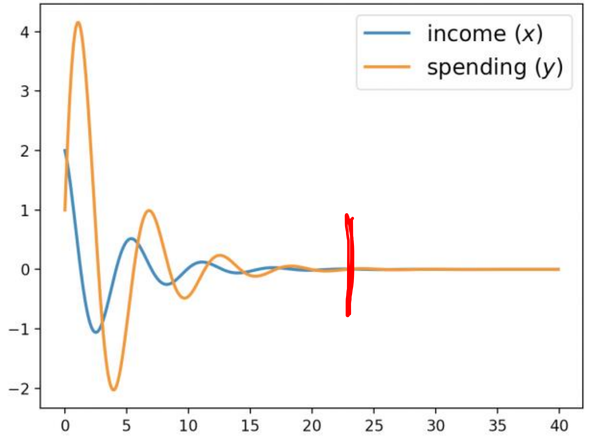
$\lambda_1 = -0.375 - i1.088$
$\lambda_2 = -0.375 + i1.088$

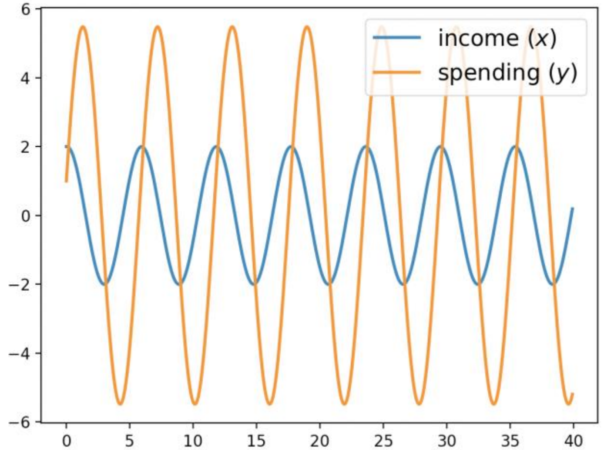


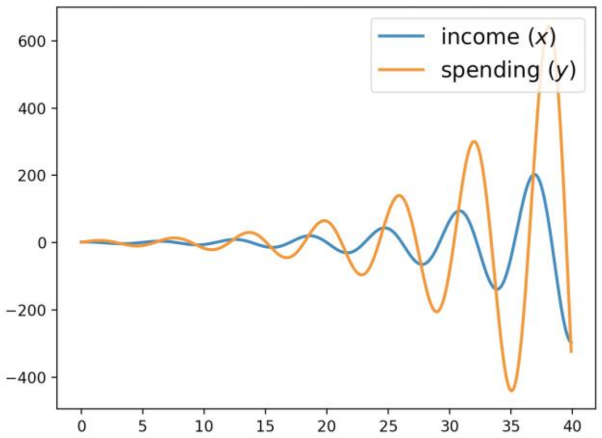


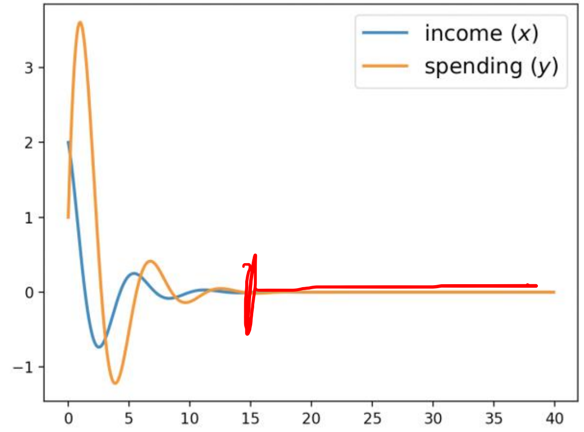




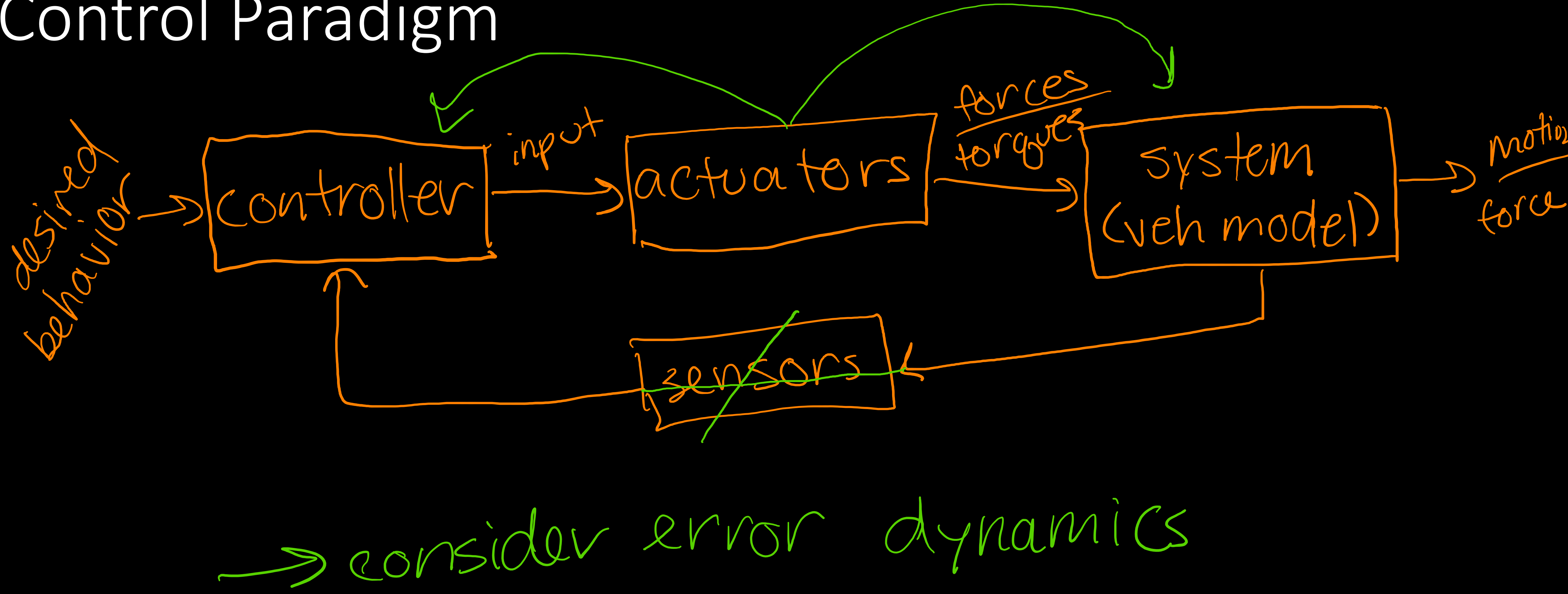




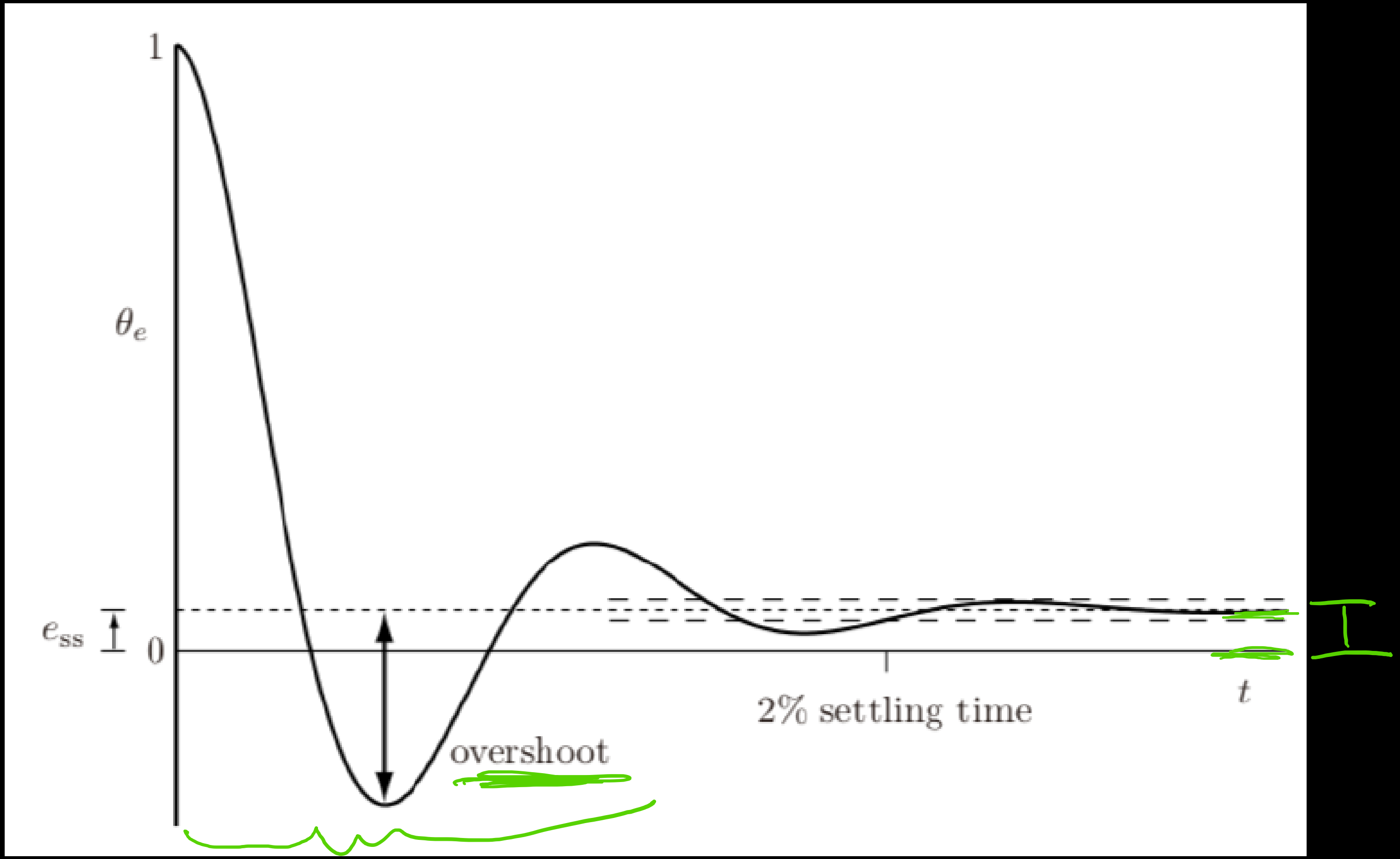




Control Paradigm



Error Dynamics



Feedback Control

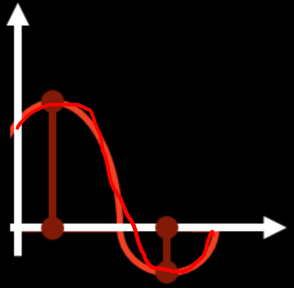
PID controller

$$u = \underbrace{K_p e}_{\text{reduce pos error}} + \underbrace{K_d \dot{e}}_{\text{dampen vel error}} + \underbrace{K_i \int e(\tau)\, d\tau}_{\text{remove ss or accumulated errors}}$$

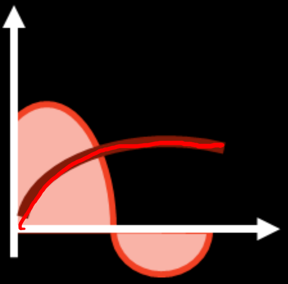
- $K_p e$ → reduce pos error
- $K_d \dot{e}$ → dampen vel error
- $K_i \int e(\tau)\, d\tau$ → remove ss or accumulated errors

PID Controllers

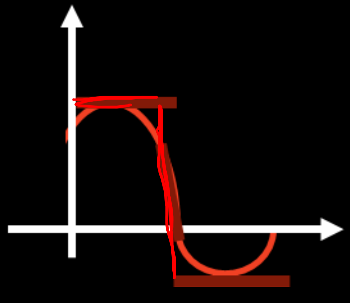
- Proportional $u = k_p e$

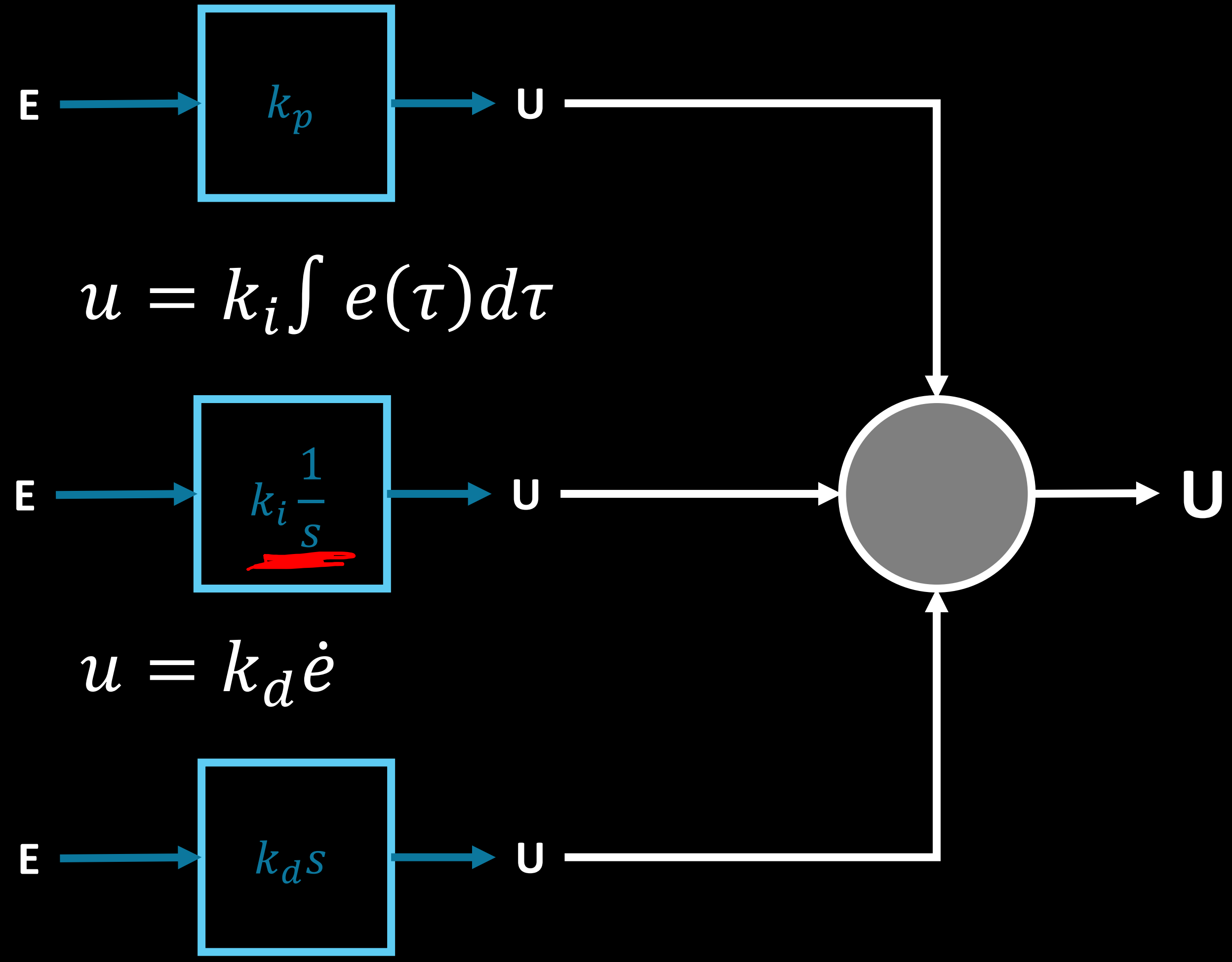


- Integral $u = k_i \int e(\tau) d\tau$



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





Linear Error Dynamics and Stability

$$a_p e^{(p)} + a_{p-1} e^{(p-1)} + \dots + a_1 \dot{e} + a_0 e = 0$$

$$x_1 = e,\quad x_2 = \dot{e},\quad x_3 = \ddot{e},\quad \dots$$

$$x_p = -\frac{a_0}{a_p} x_1 - \frac{a_1}{a_p} x_2 - \dots$$

$$\dot{x} = Ax,\quad x = [x_1 \dots x_p]^T$$

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

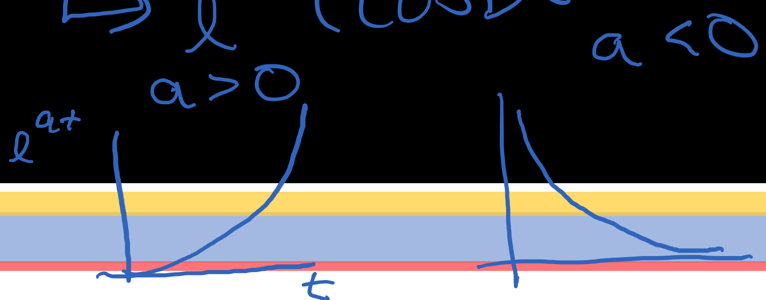
recall: $\dot{x} = Ax \rightsquigarrow x(t) = e^{At} x_0$

sys is stable if the real parts of the eigenvalues are neg.

$\rightarrow$ unstable if $\exists\, x_0$ s.t. $\lim_{t \to \infty} \|x(t)\| = \infty$

e^{dt}, where $d = a + bi \in \mathbb{C}$

$\hookrightarrow e^{at}(\cos bt + i \sin bt)$



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:

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

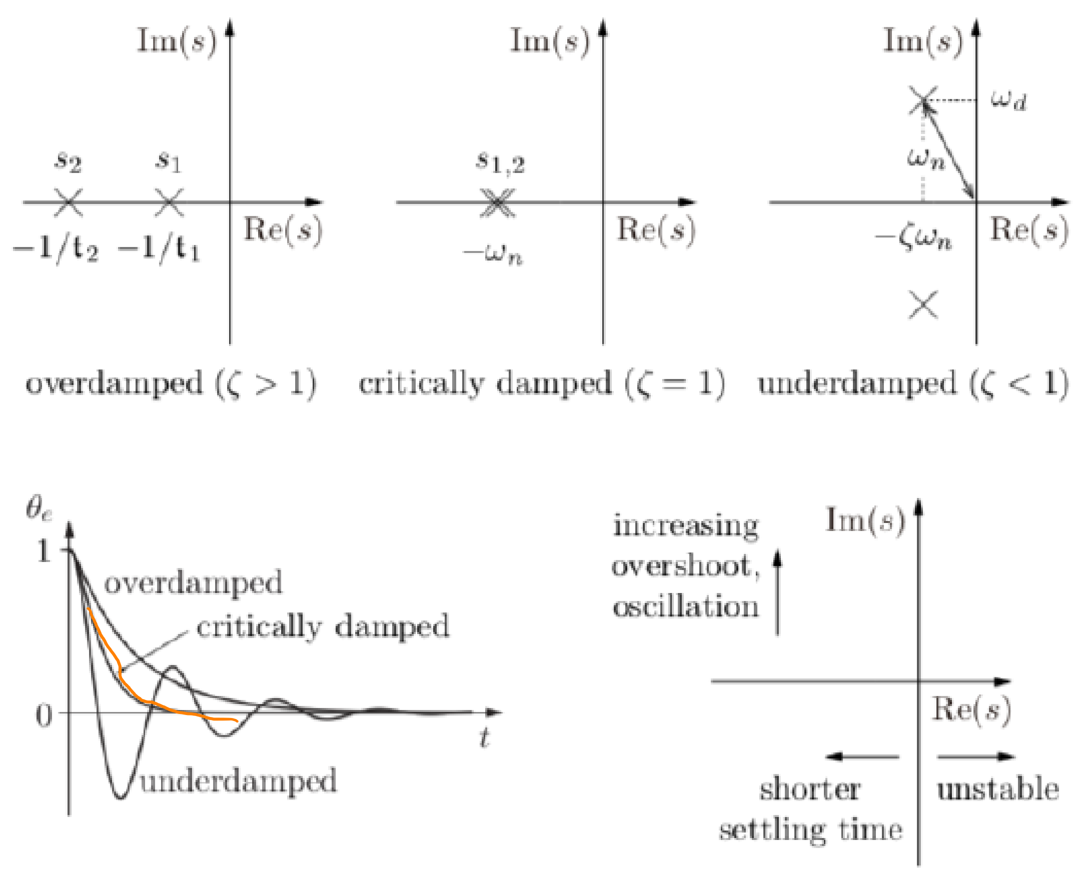
- Note that the system is stable iff ω_n and ξ are positive

Second Order Dynamics: Cases

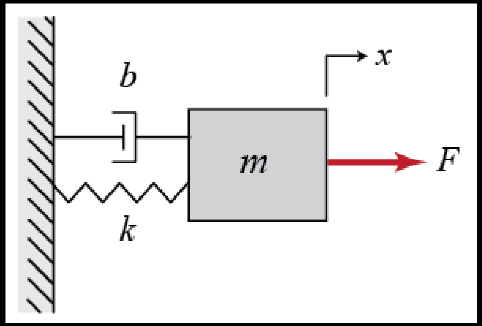
- Overdamped: $\zeta > 1$
  - Roots s_1 and s_2 are distinct
  - $\theta_e(t) = c_1 e^{s_1 t} + c_2 e^{s_2 t}$
  - Time constant is the less negative root
- Critically damped: $\zeta = 1$
  - Roots s_1 and s_2 are equal and real
  - $\theta_e(t) = (c_1 + c_2 t)e^{-\omega_n t}$
  - Time constant is given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

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

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



Simple Damped Spring System



$$m\,\ddot{x} + b\dot{x} + kx = F$$

$$\ddot{x} + \frac{b}{m}\dot{x} + \frac{k}{m}x = u$$

$$\ddot{x} + 2\xi\omega_0\dot{x} + \omega_0^2 x = u$$

ξ damping ratio

ω_0 natural frequency

$$m\,\ddot{x} + b\dot{x} + kx = u$$

$$\mathcal{L}\{m\ddot{x} + b\dot{x} + kx\} =$$

$$ms^2X(s) + bsX(s) + kX(s)$$

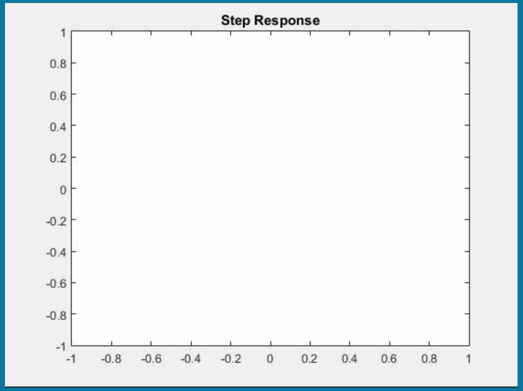
Transfer Function:

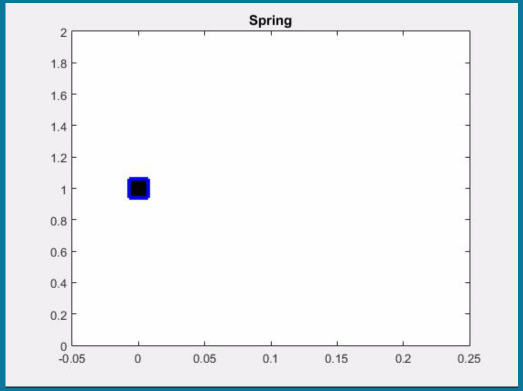
$$\frac{X(s)}{U(s)} = \frac{1}{ms^2 + bs + k}$$

Poles:

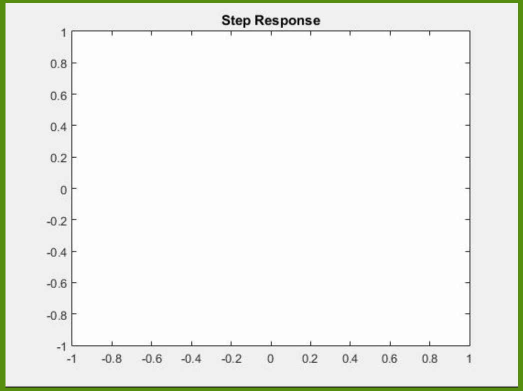
$$s = \frac{-b \pm \sqrt{b^2 - 4mk}}{2m}$$

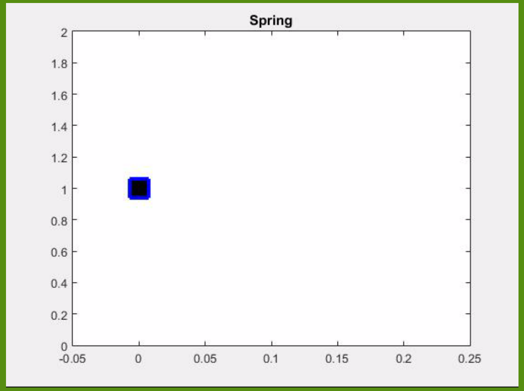
Undamped Case: $b = 0$



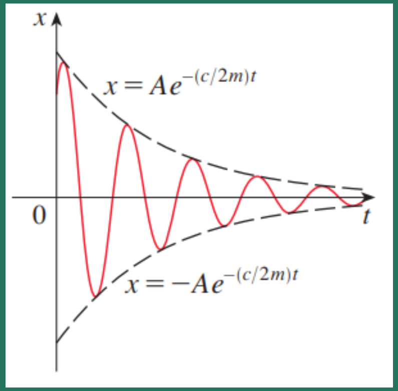


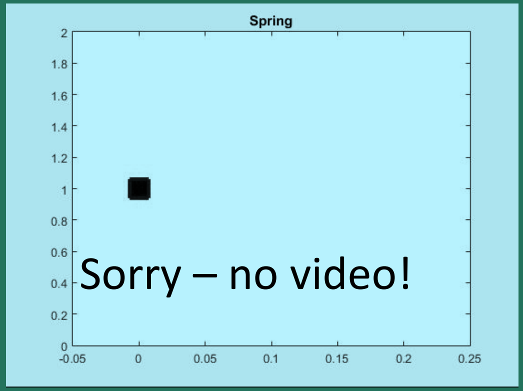
Overdamped Case: $b^2 - 4mk > 0$



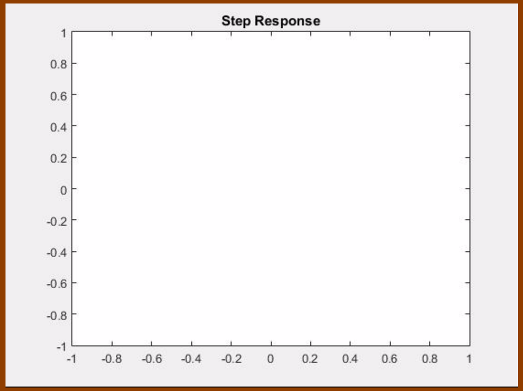


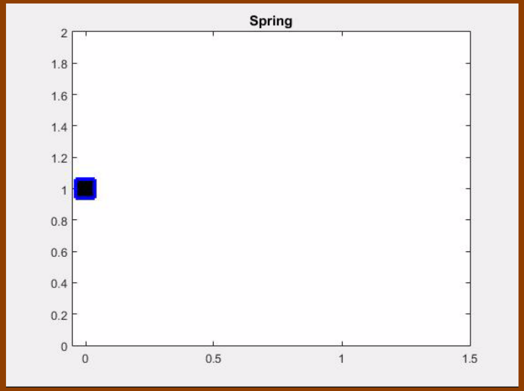
Underdamped Case: $b^2 - 4mk < 0$





With Feedback Control

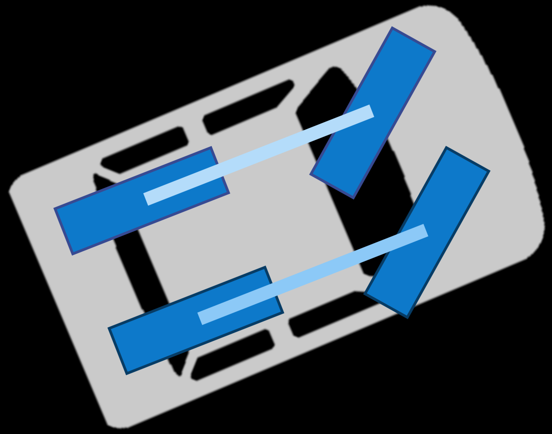




Break!

Simple vehicle model: Dubin's car

- Key assumptions
  - Front and rear wheel in the plane in a stationary coordinate system
  - Steering input, front wheel steering angle δ
  - No slip: wheels move only in the direction of the plane they reside in
- Zeroing out the accelerations perpendicular to the plane in which the wheels reside, we can derive simple equations



Reference: Paden, Brian, Michal Cap, Sze Zheng Yong, Dmitry S. Yershov, and Emilio Frazzoli. 2016. A survey of motion planning and control techniques for self-driving urban vehicles. IEEE Transactions on Intelligent Vehicles 1 (1): 33–55.

Dubin's Model

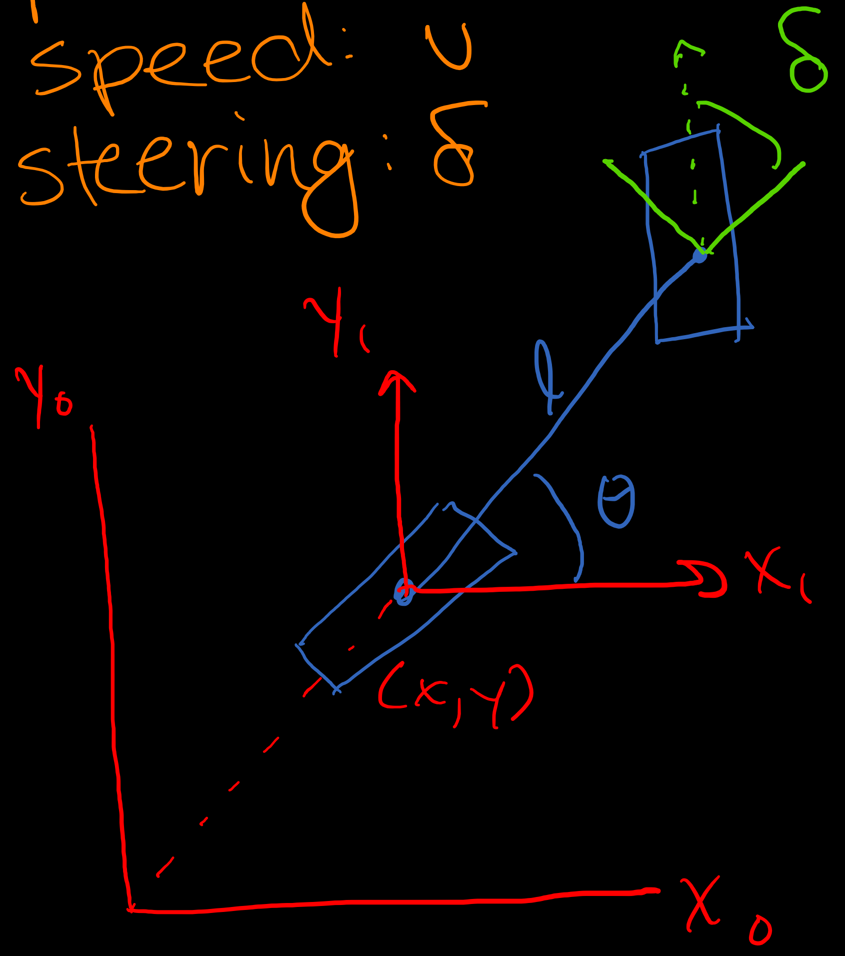
$$\begin{cases} \dot{x} = v \cos\theta \\ \dot{y} = v \sin\theta \\ \dot{\theta} = \frac{v}{\ell} \tan\delta \end{cases}$$

length ℓ

pose : $[x, y, \theta]^T$

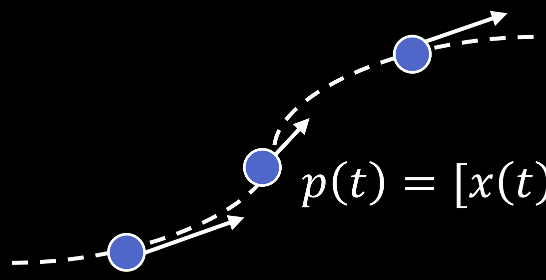
speed : v

steering : δ



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
- 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:

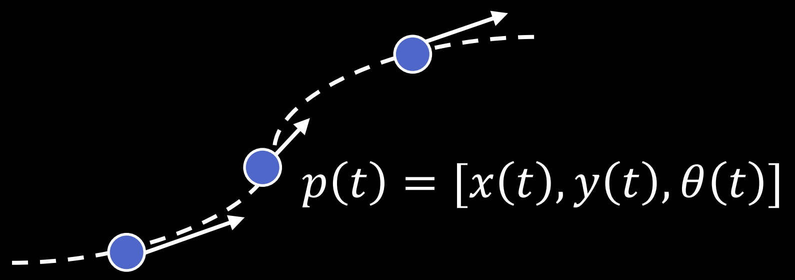
$$u_{\omega,OL}(t) = \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} = \begin{bmatrix} \sqrt{\dot{x}(t)^2 + \dot{y}(t)^2} \\ \dot{\theta}(t) \end{bmatrix}$$

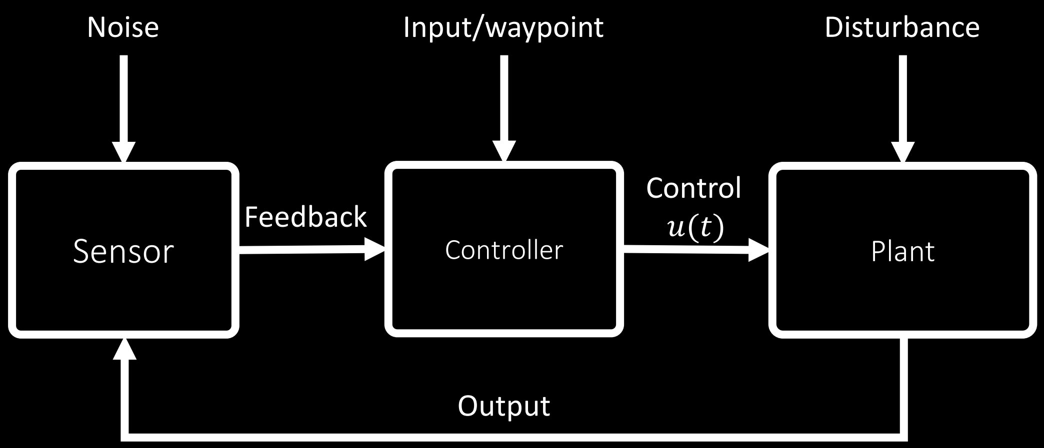
- We can write an open-loop controller for a robot with car-like steering:

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

Path following control

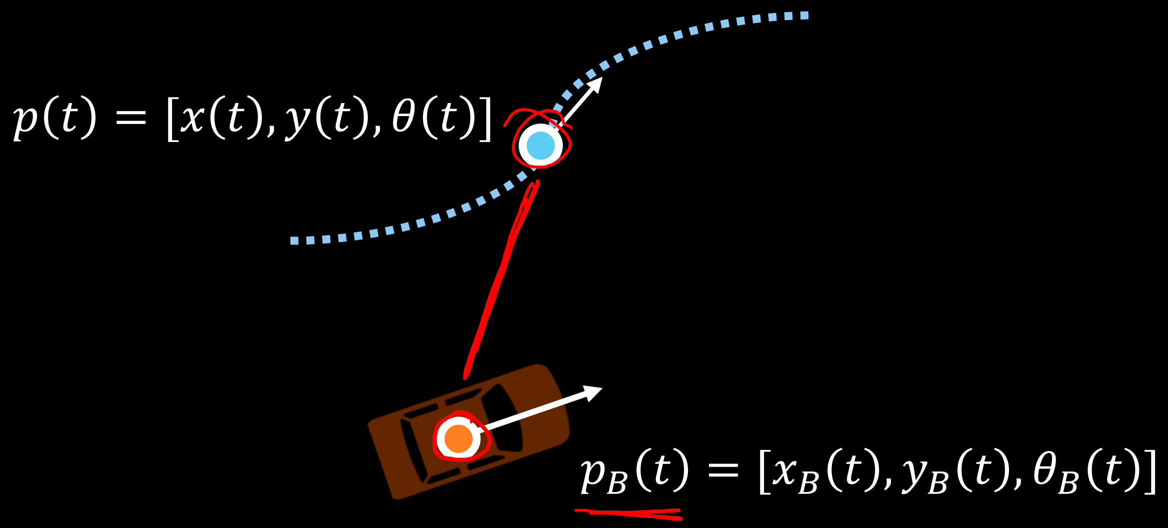
- The path followed by a robot can be represented by a *trajectory or path* parameterized by time
  - → from a higher-level planner
- Defines the desired instantaneous pose $p(t)$





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

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

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

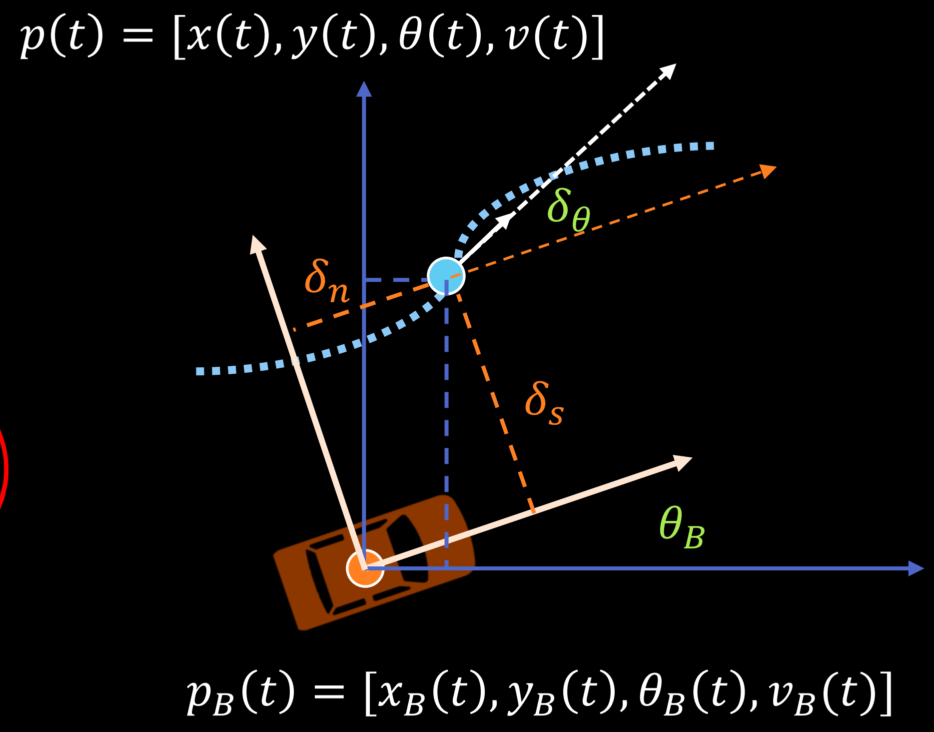
- **Cross track error:** portion of the position error orthogonal to the intended direction of motion

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

- **Heading error:** difference between desired and actual orientation and direction

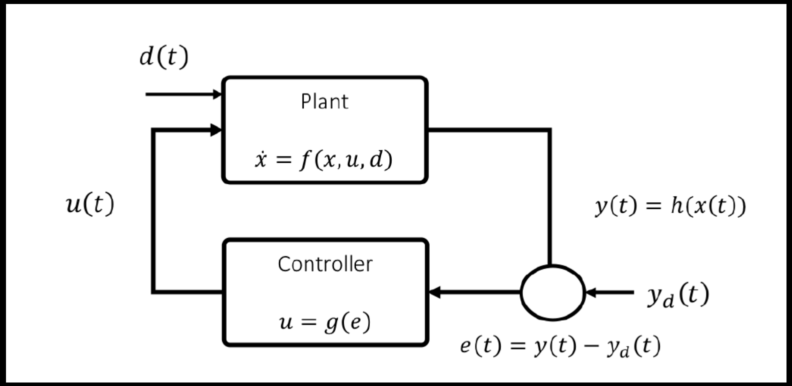
$$\delta_\theta = \theta(t) - \theta_B(t)$$

$$\delta_v = v(t) - v_B(t)$$



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

A simple P-controller example



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

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

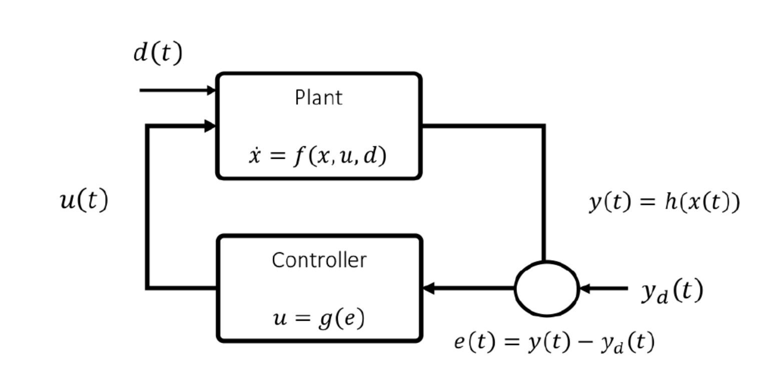
$$\dot{y}(t) = -K_P y(t) + K_P y_d(t) + d(t)$$

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

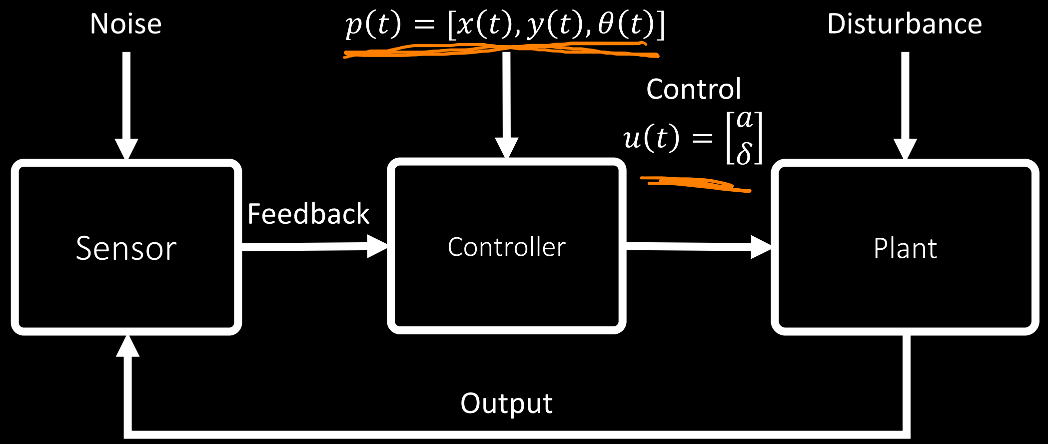
Control Law

Control input is given by $u = [a, \delta]^{\mathsf{T}}$

where a is the acceleration and δ is the steering angle

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

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

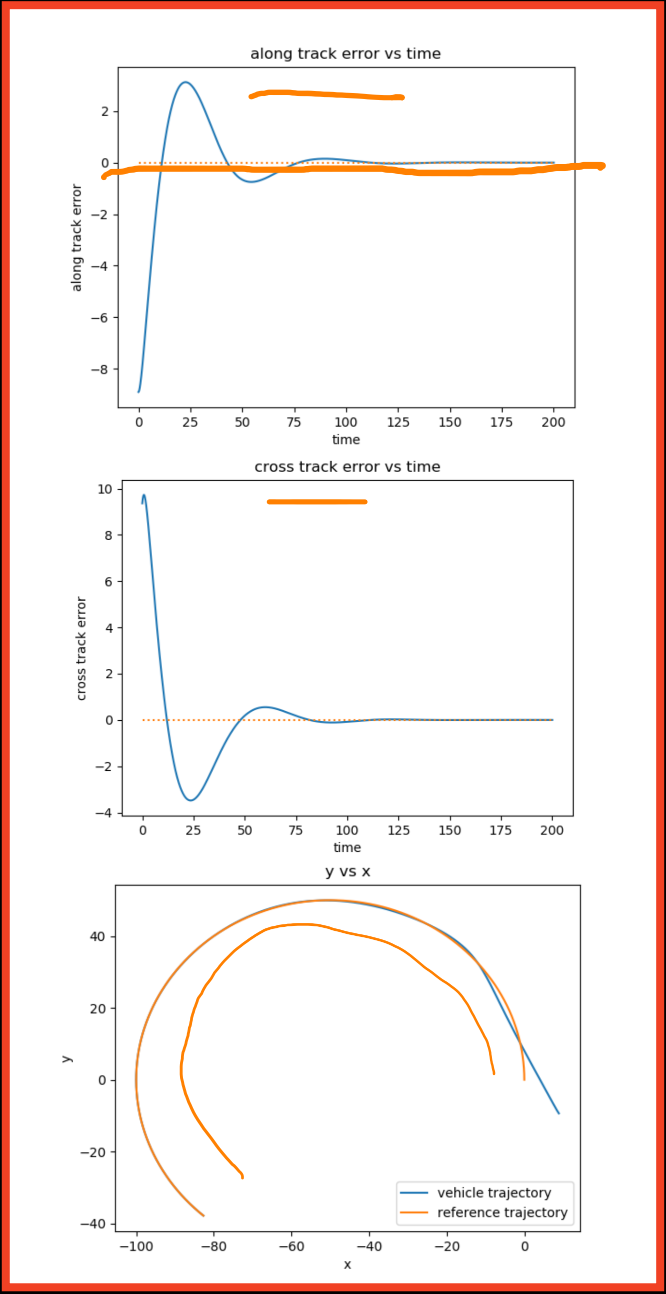


Control Law

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

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.

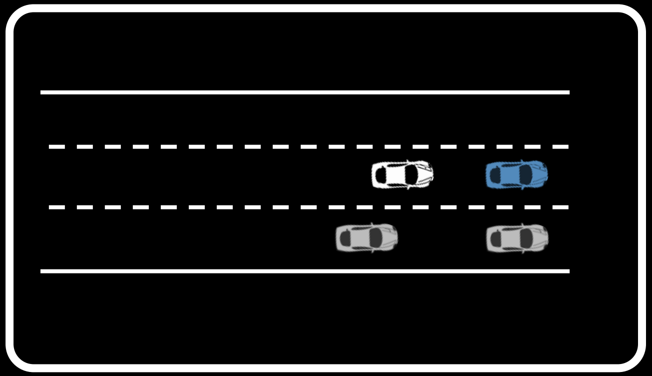


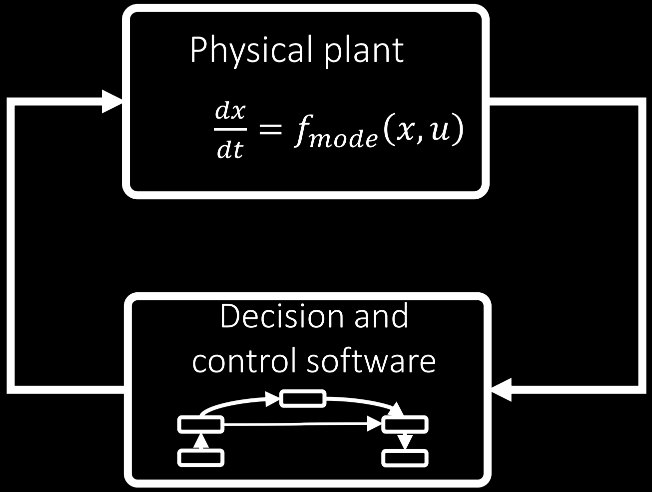
Summary

- 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 MP2! Tutorial on 3/11.
- *Next time:* Advanced Control Topics!

Extra Slides

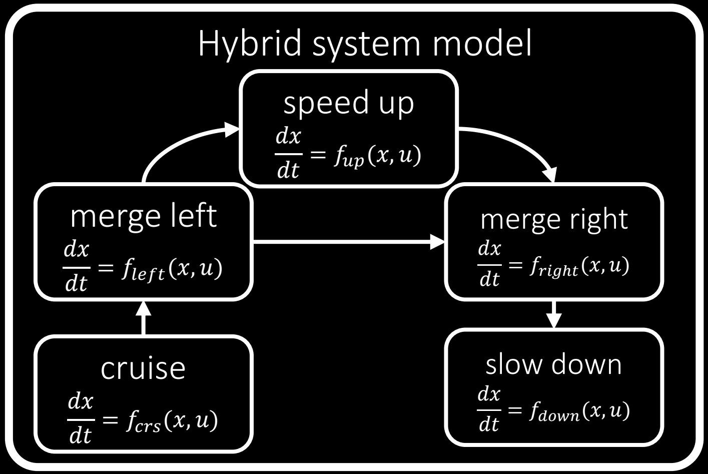
Typical system models





Nonlinear *hybrid* dynamics

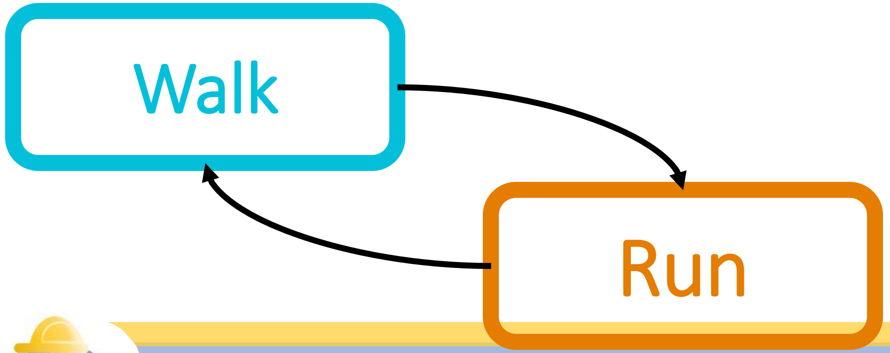
Interaction between computation and physics can lead to unexpected behaviors

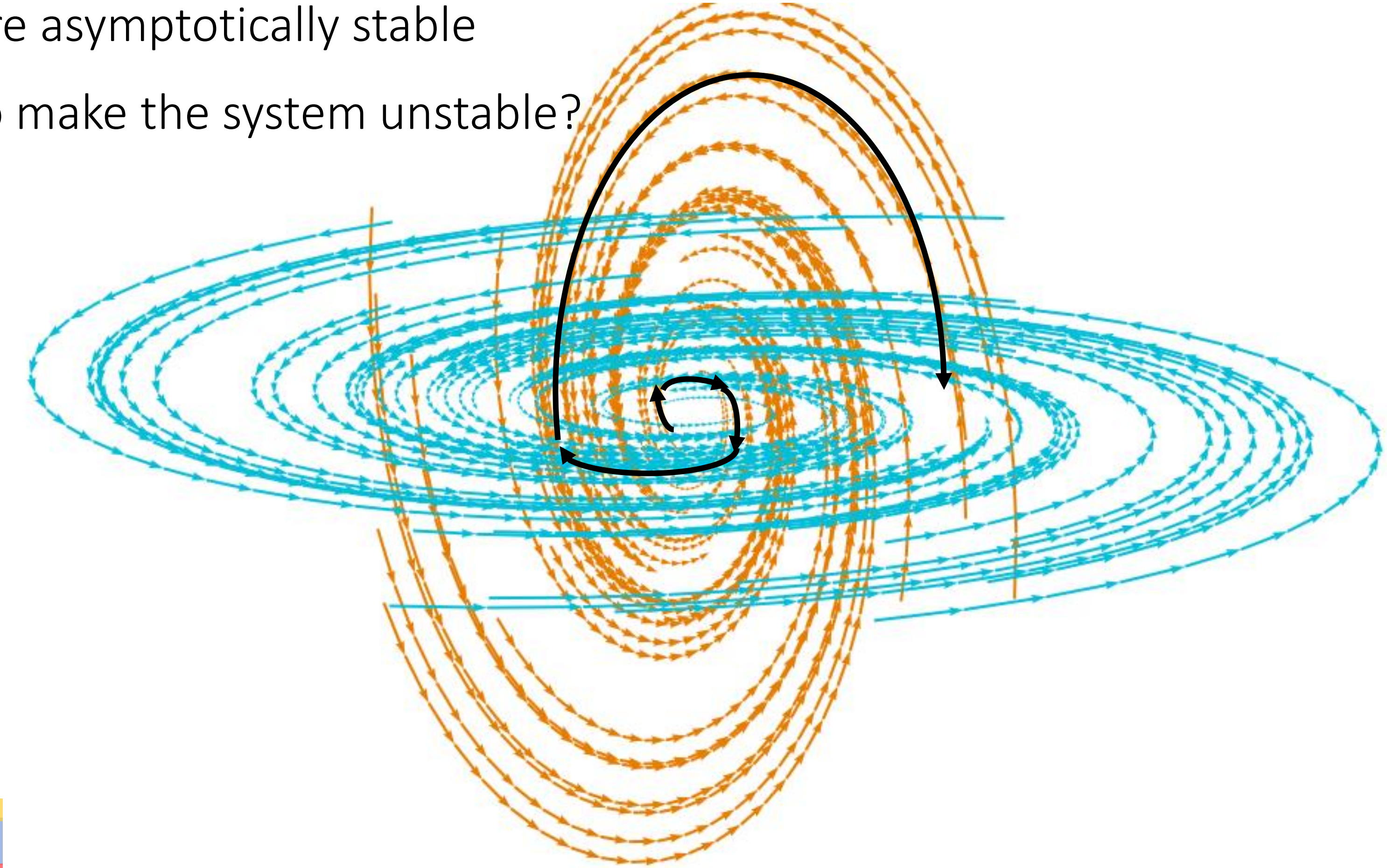


Hybrid Instability: Switching between two stable linear models

Each of the modes of a walking robot are asymptotically stable

Is it possible to switch between them to make the system unstable?





Yes! By switching between them the system becomes unstable