

# Lecture 10: Control II

Professor Katie Driggs-Campbell

February 20, 2024

ECE484: Principles of Safe Autonomy



# Administrivia

- Team formation due this week
- 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
- Bonus MP in April



# Today's Plan

- Review some differential equations and linear algebra
- Take a look at PID controllers
- Build up waypoint following using the models discussed previously

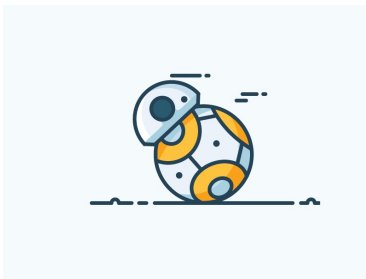


# Dynamical Systems Model

Describe behavior in terms of instantaneous laws:

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

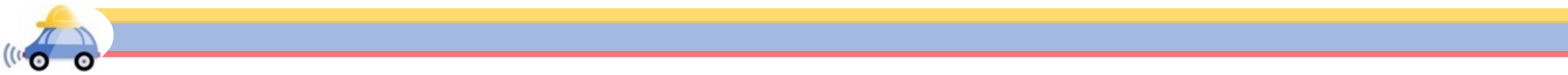
where  $t \in \mathbb{R}$ ,  $x(t) \in \mathbb{R}^n$ ,  $u(t) \in \mathbb{R}^m$ , and  $f: \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$  gives the dynamics / transition function



# Recall (1)



# Recall (2)

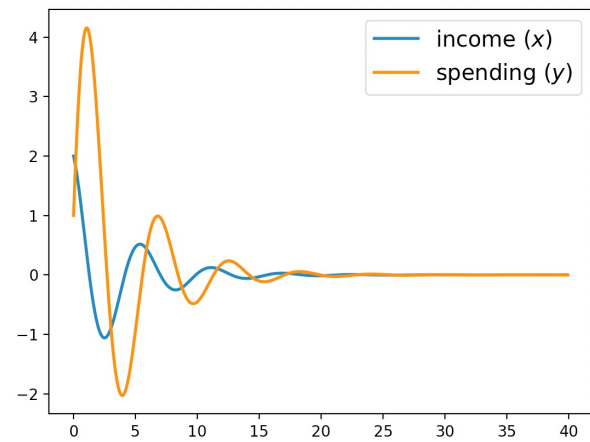
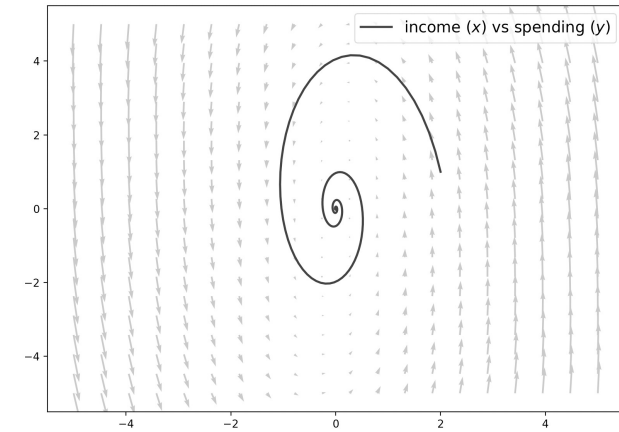


# Examples

$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} -1/4 & -2/5 \\ 3 & -1/4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

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

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

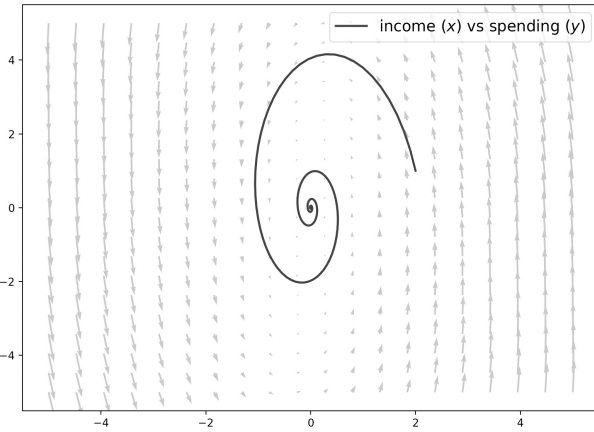


# Examples

$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} -1/4 & -2/5 \\ 3 & -1/4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

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

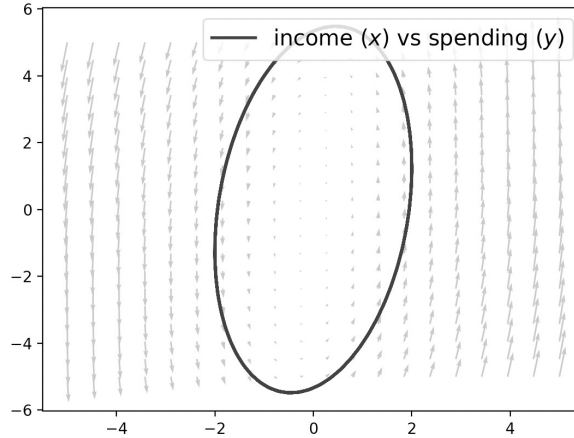
$$\lambda_2 = -0.25 + i1.10$$



$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} 1/4 & -2/5 \\ 3 & -1/4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\lambda_1 = +i0.1066$$

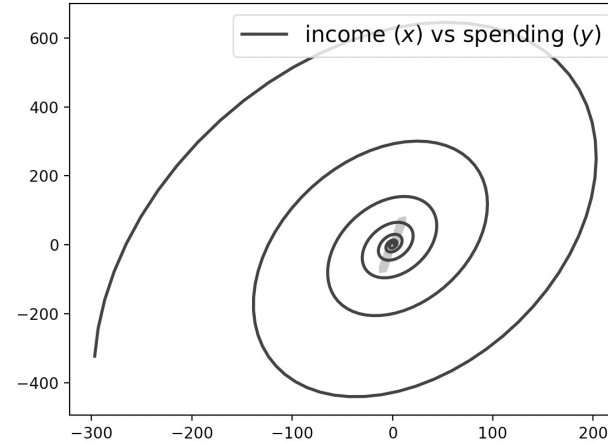
$$\lambda_2 = -i0.1066$$



$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} 1/2 & -2/5 \\ 3 & -1/4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

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

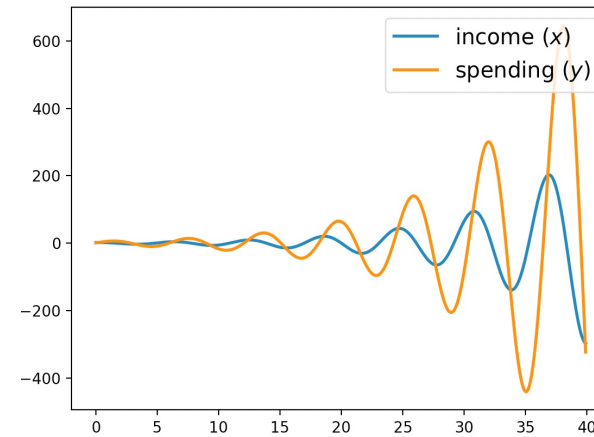
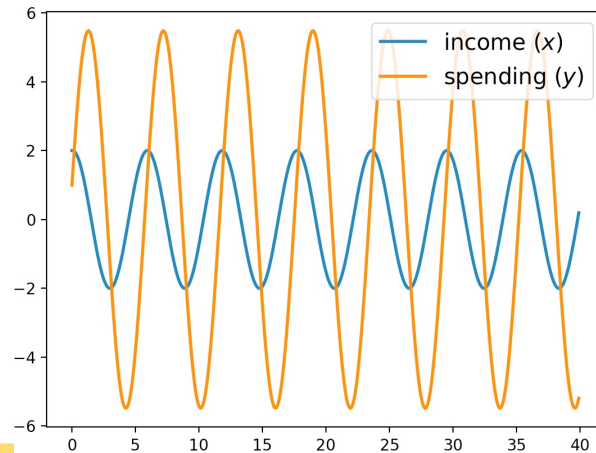
$$\lambda_2 = -0.125 - i1.029$$



$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} 1/2 & -2/5 \\ 3 & -1/4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

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

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



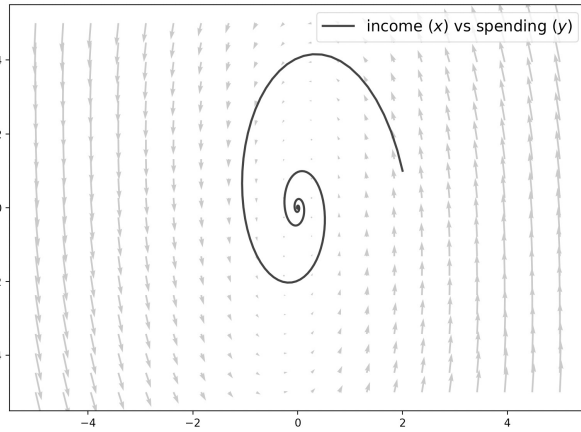


# Examples

$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} -1/4 & -2/5 \\ 3 & -1/4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

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

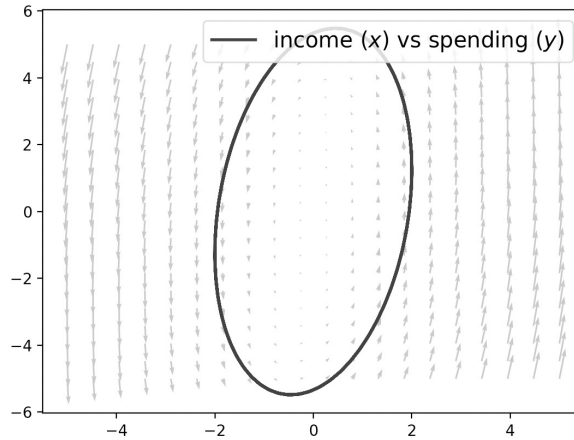
$$\lambda_2 = -0.25 + i1.10$$



$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} 1/4 & -2/5 \\ 3 & -1/4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\lambda_1 = +i0.1066$$

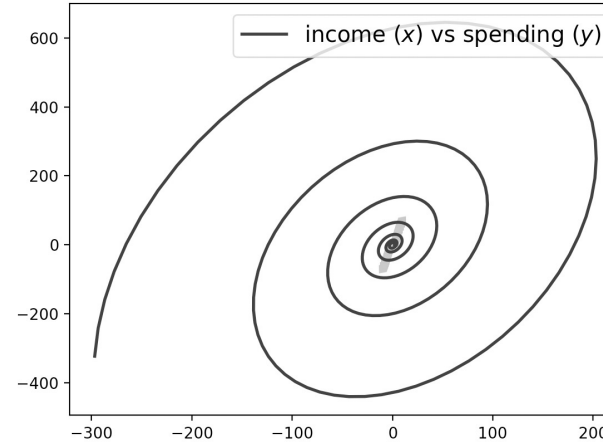
$$\lambda_2 = -i0.1066$$



$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} 1/2 & -2/5 \\ 3 & -1/4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

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

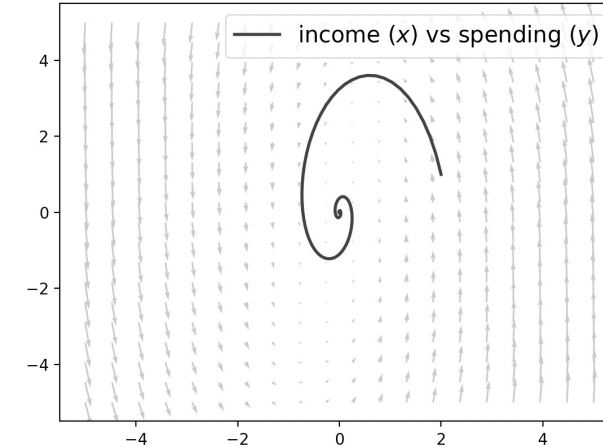
$$\lambda_2 = -0.125 - i1.029$$



$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} -1/4 & -2/5 \\ 3 & -1/2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

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

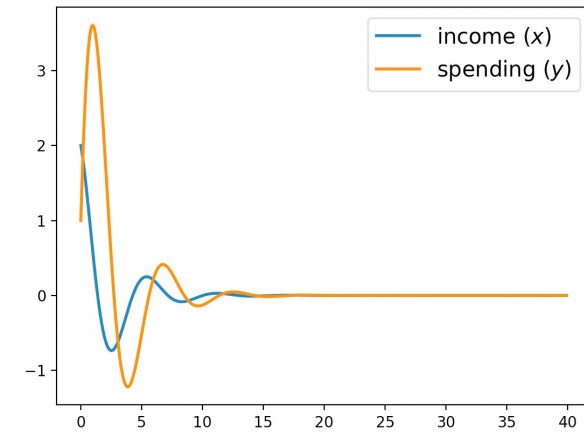
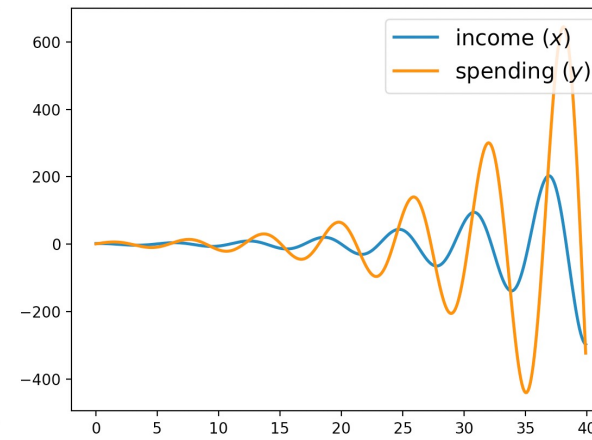
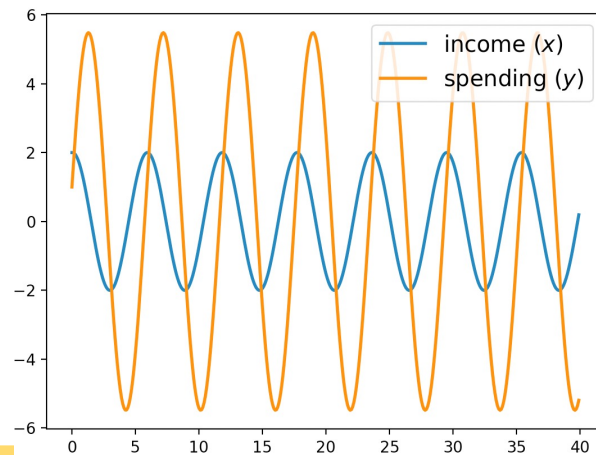
$$\lambda_2 = -0.375 + i1.088$$



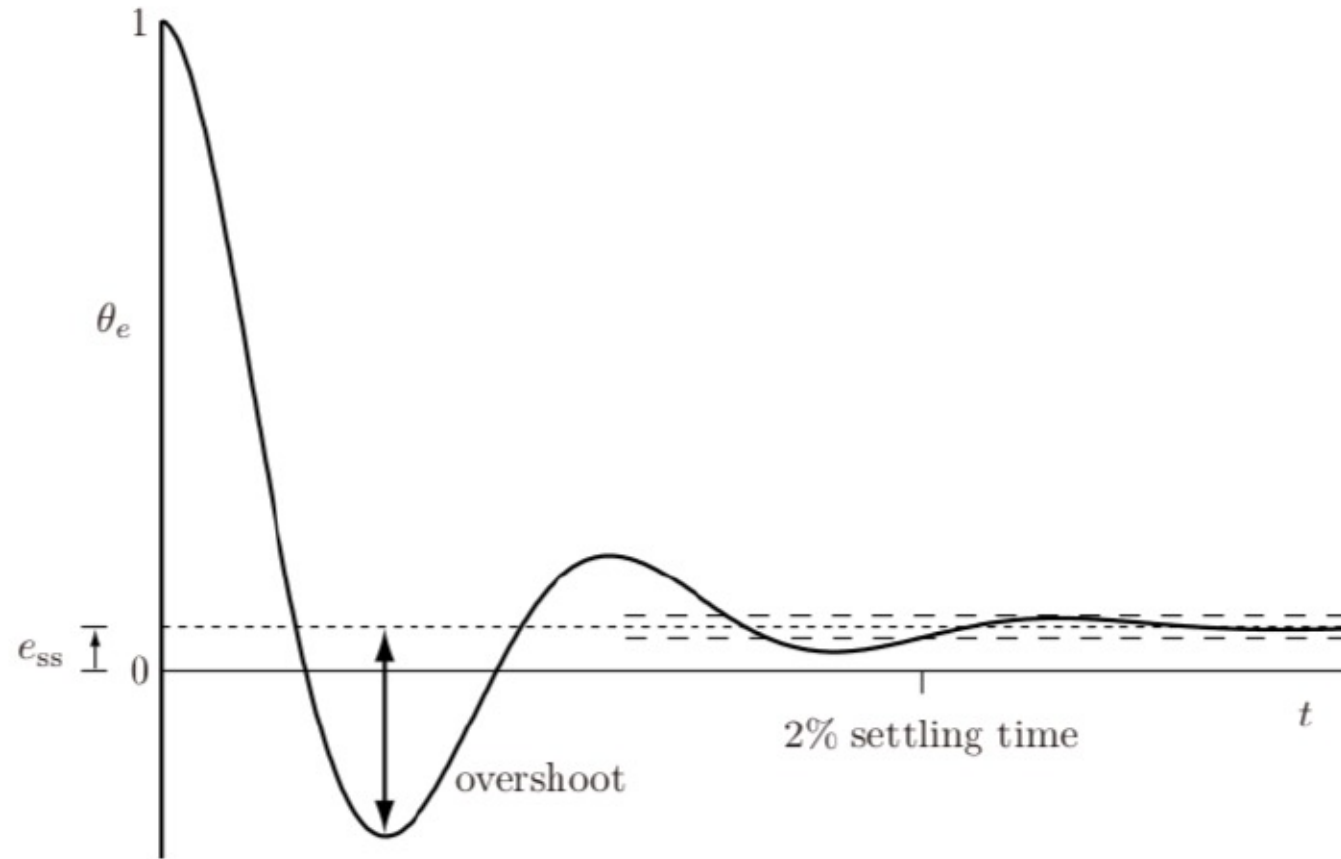
$$\begin{bmatrix} \dot{x}_2 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} 1/2 & -2/5 \\ 3 & -1/4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

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

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



# Error Dynamics

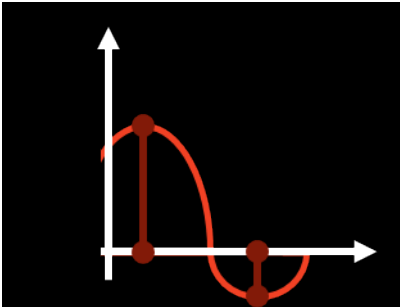


# Feedback Control

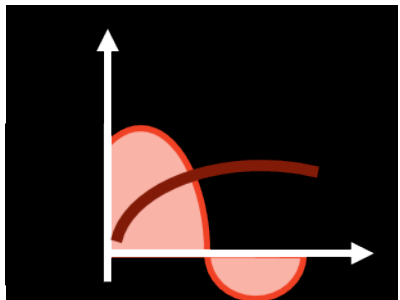


# PID Controllers

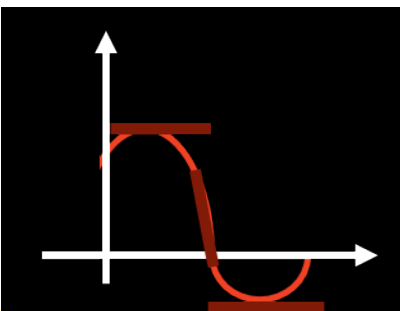
- Proportional



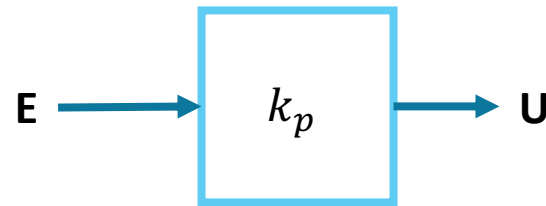
- Integral



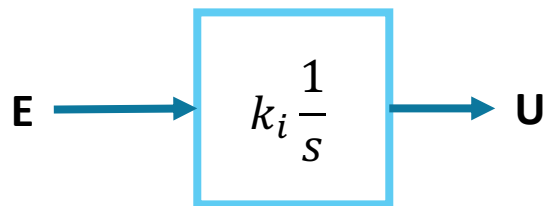
- Derivative



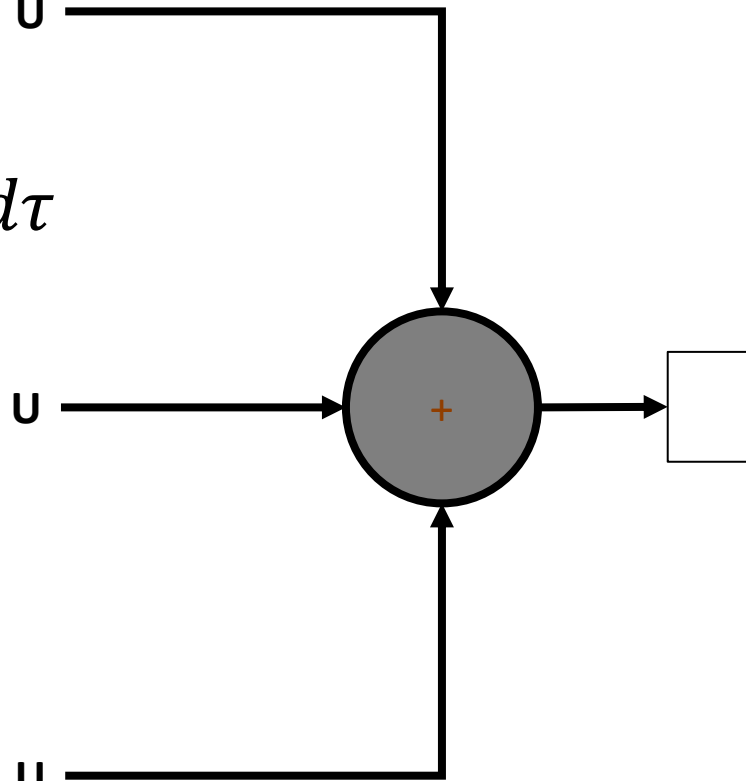
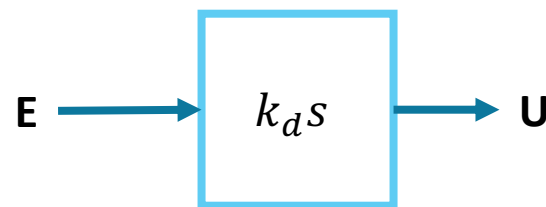
$$u = k_p e$$



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



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



# Linear Error Dynamics



# Viewing as a Second Order System

- The second order system is:  $\ddot{e} + c_1\dot{e} + c_2e = 0$
- In standard form, we write:

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

where  $\xi$  is the *damping ratio* and  $\omega_n$  is the *natural frequency*

- The eigenvalues are given as:

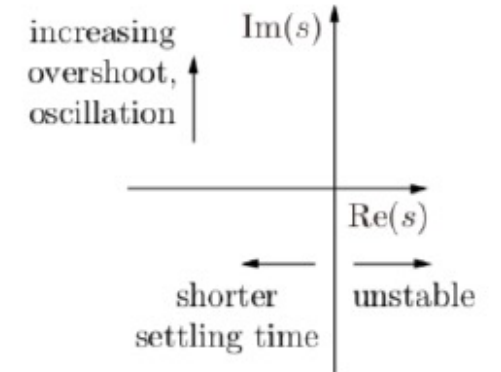
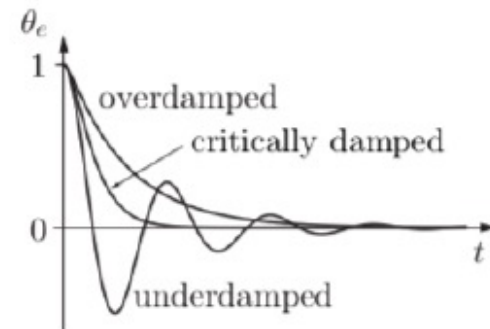
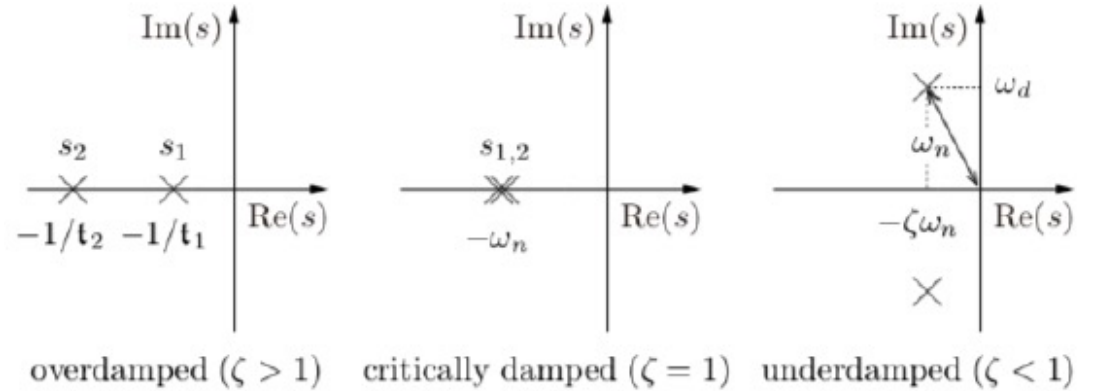
$$\lambda_{1,2} = -\xi\omega_n \pm \omega_n\sqrt{\xi^2 - 1}$$

- Note that the system is stable iff  $\omega_n$  and  $\xi$  are positive

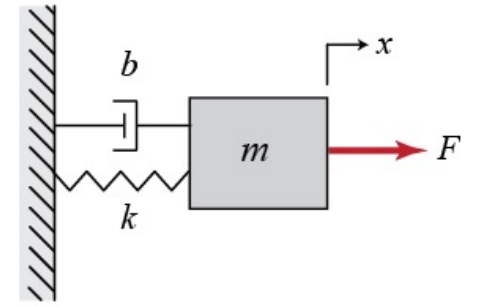


# Second Order Dynamics: Cases

- 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



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

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

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

$\xi$  damping ratio

$\omega_0$  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:

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

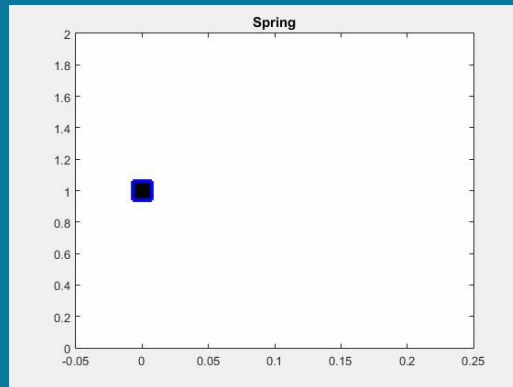
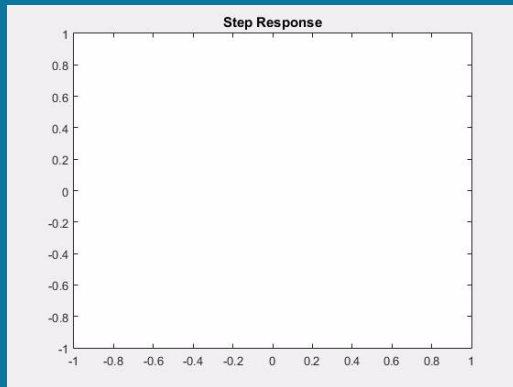
Poles:

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

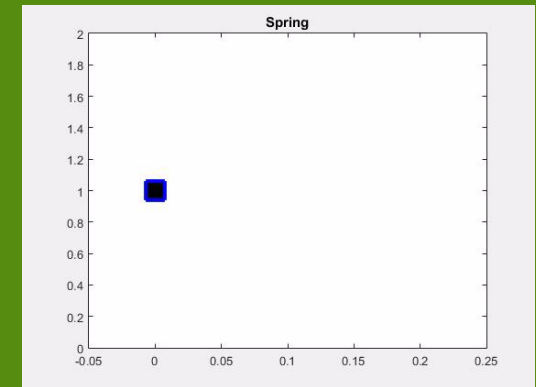
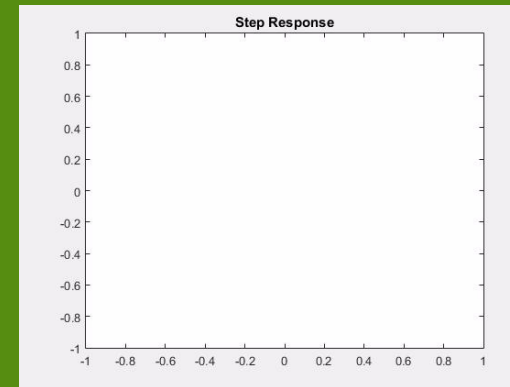




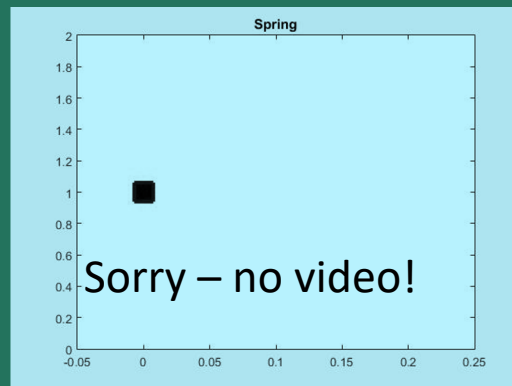
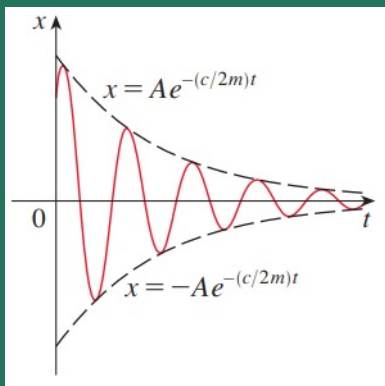
## Undamped Case: $b = 0$



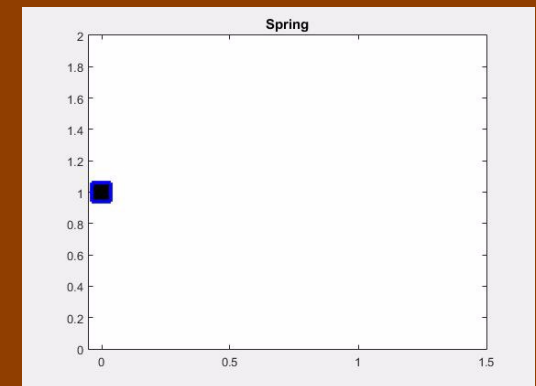
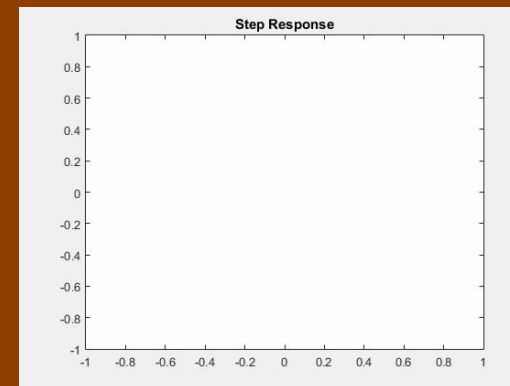
## Overdamped Case: $b^2 - 4mk > 0$



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



## With Feedback Control

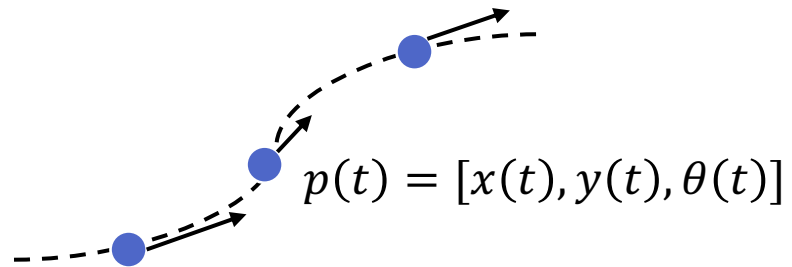


On to PID for path following



# 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  $p(t)$



# 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:

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

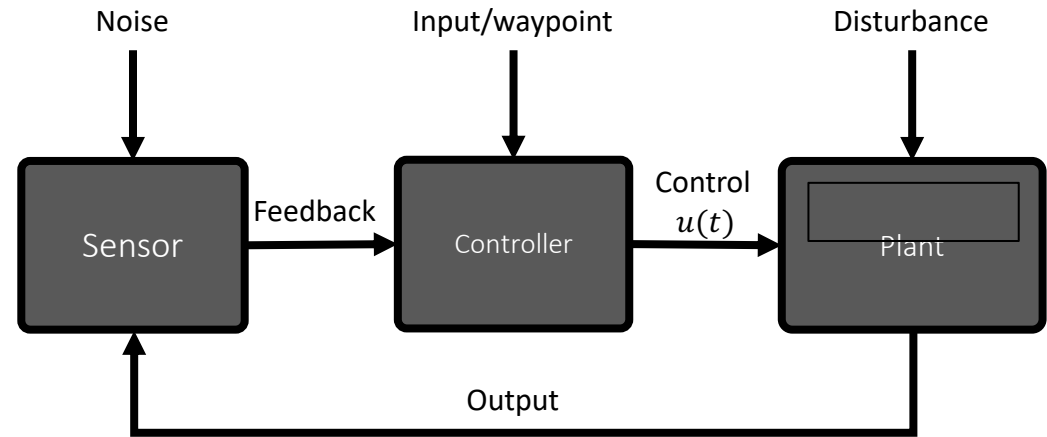
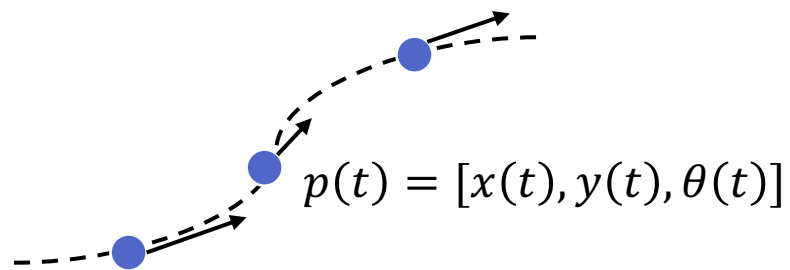
- We can write an **open-loop controller** for a robot with car-like steering:

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



# Path following control

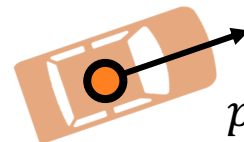
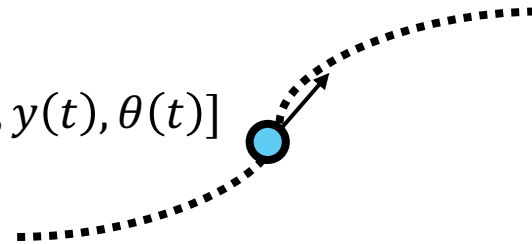
- The path followed by a robot can be represented by a *trajectory or path* parameterized by time
  - from a higher-level planner, map, or perception system
- Defines the desired instantaneous pose  $p(t)$



# 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  $y_d(t) - y(t)$ ?

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



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



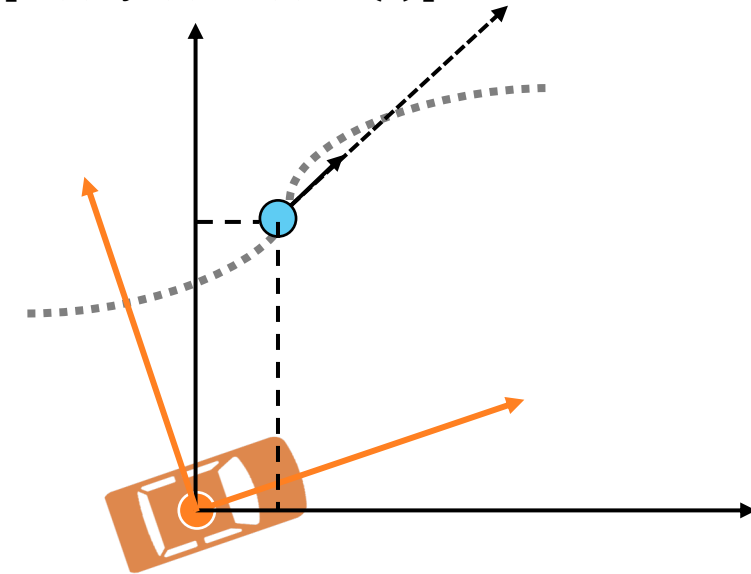
# Path following control

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

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



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



# 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

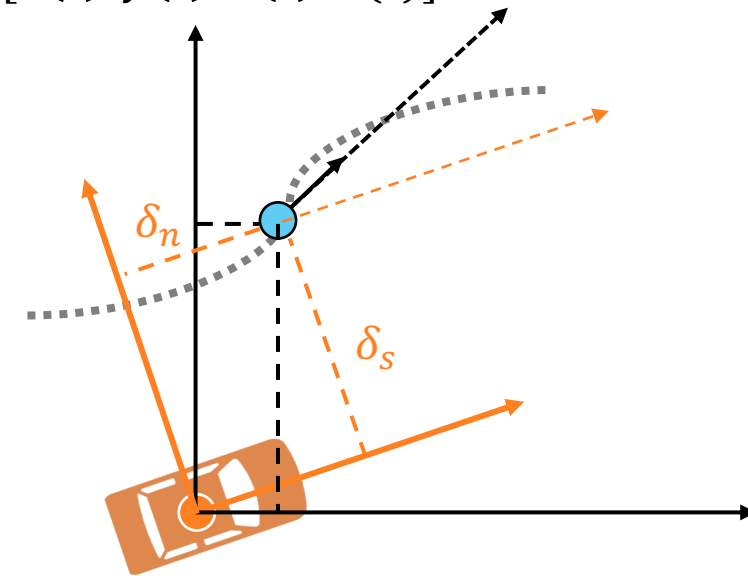
- **Along track error:** distance ahead or behind the target in the instantaneous direction of motion.

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

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

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

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



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





# 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

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

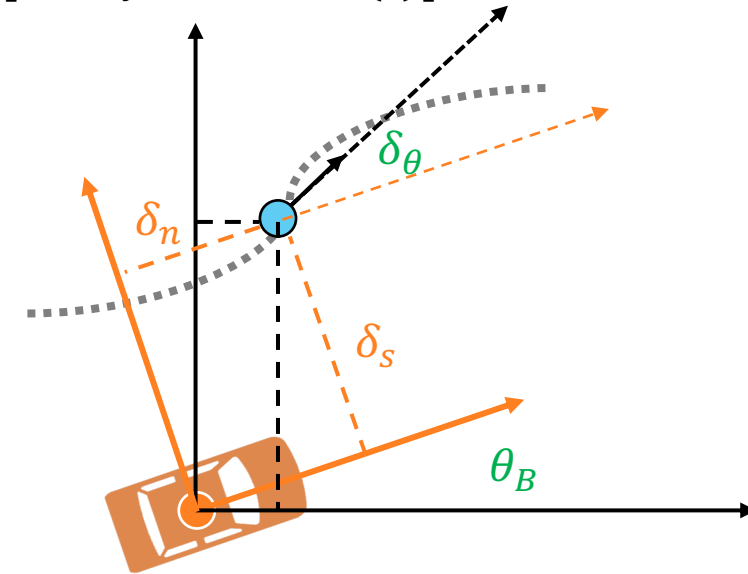
- **Heading error:** difference between desired and actual orientation and direction

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

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

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

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



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

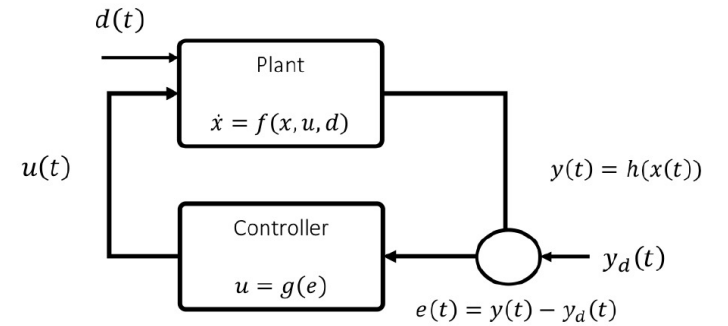


# A simple P-controller example

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

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

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



# A simple P-controller example

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

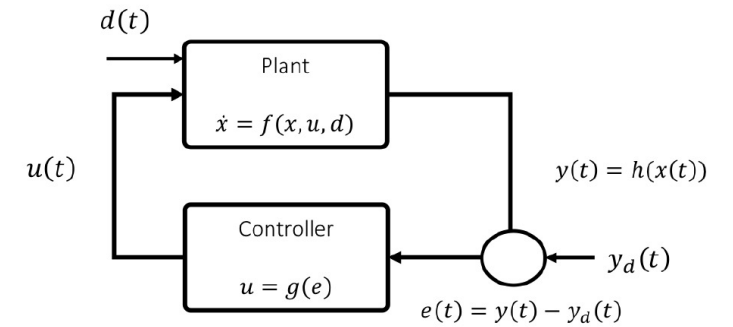
$$u(t) = -K_P e(t) = -K_P (y(t) - y_d(t))$$

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

- Consider constant setpoint  $y_0$  and disturbance  $d_{ss}$

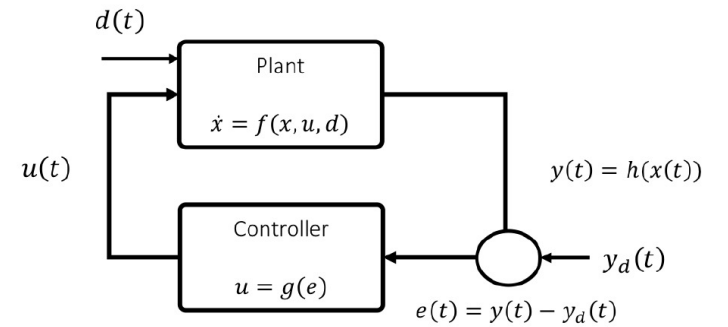
$$\dot{y}(t) = -K_P y(t) + K_P y_0 + d_{ss}$$

- 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:  $\dot{y}(t) = u(t) + d(t)$
- 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  $y_{ss}$ :  $y(t) = \frac{d_{ss}}{K_P} + y_0$

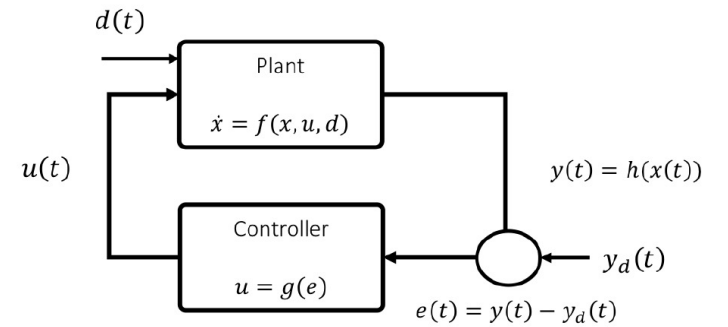


# A simple P-controller example

- 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  $d_{ss}$

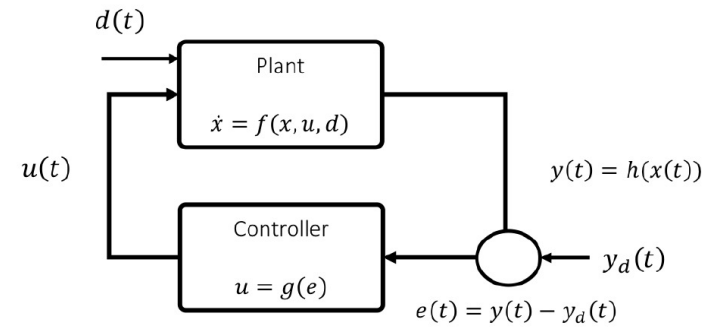
$$\dot{y}(t) = -K_P y(t) + K_P y_0 + d_{ss}$$

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

- Transient behavior:

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

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

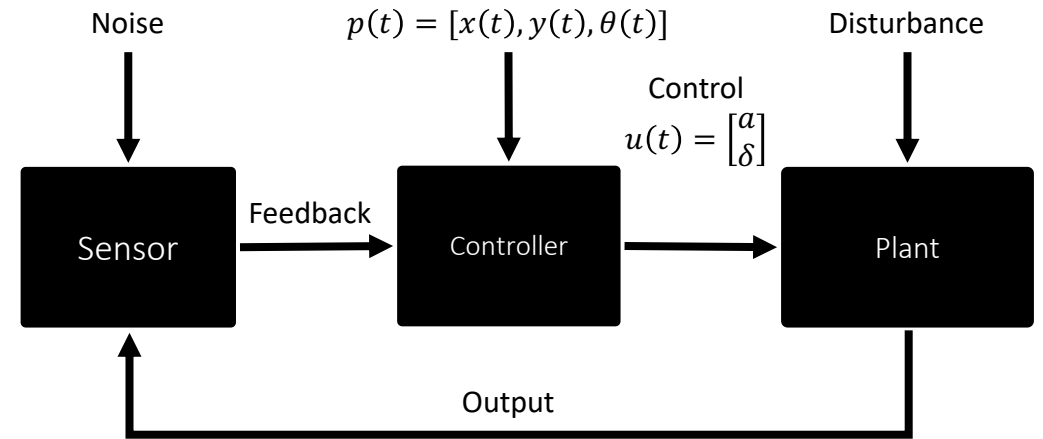


# Control Law

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

where  $a$  is the acceleration and  $\delta$  is the steering angle

$$u = K \begin{bmatrix} \delta_s \\ \delta_n \\ \delta_\theta \\ \delta_v \end{bmatrix}$$
$$K = \begin{bmatrix} K_s & 0 & 0 & K_v \\ 0 & K_n & K_\theta & 0 \end{bmatrix}$$

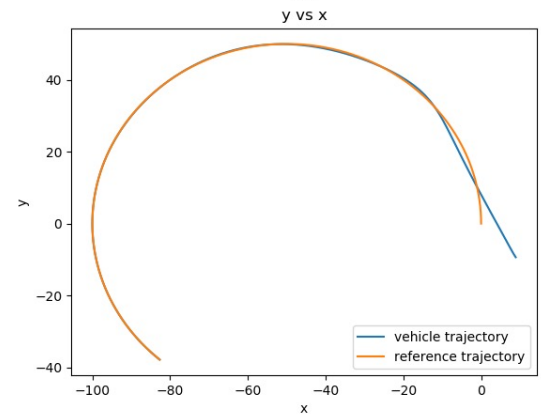
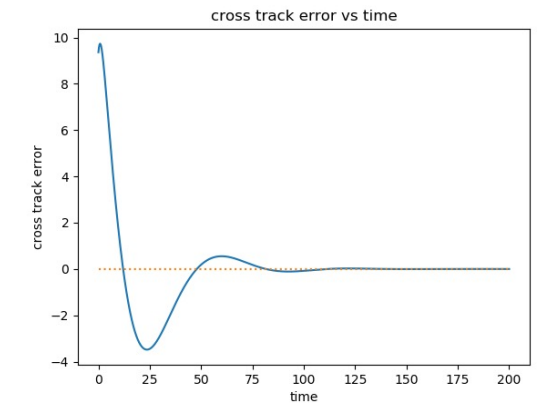
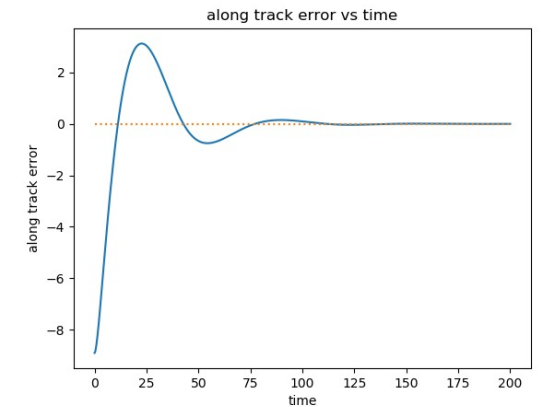


# Control Law

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

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  $\delta_\theta$  is related to the derivative of  $\delta_n$ .





# 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 needed for MP2!
- *Next time: Advanced Control Topics!*

