

ECE 4160/5160
MAE 4910/5910

Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

Fast Robots

PID (continued)

Logistics

- Please answer workload polls to help students next year!

The screenshot shows a web browser window displaying an Ed Discussion thread. The browser's address bar shows the URL `edstem.org/us/courses/33676/discussion/2484781`. The page title is "ECE4160/ECE5160/MAE4910/MAE5910 - Ed Discussion". The thread is titled "Workload? #16" and is posted by Kirstin Petersen, a staff member, 2 weeks ago. The thread contains a poll with the following text: "How long did it take you to complete lab 1, ... include pre-lab, lab, and write-up? (We will use this data only to inform improvements of class in future years)". The poll results are as follows:

Response	Percentage
2-4hrs	72%
4-6hrs	16%
6-8hrs	4%
8hrs+	8%

The poll has 25 votes. The thread also shows a search bar, a filter dropdown, and a list of other threads from the same course.

ECE 4160/5160
MAE 4910/5910

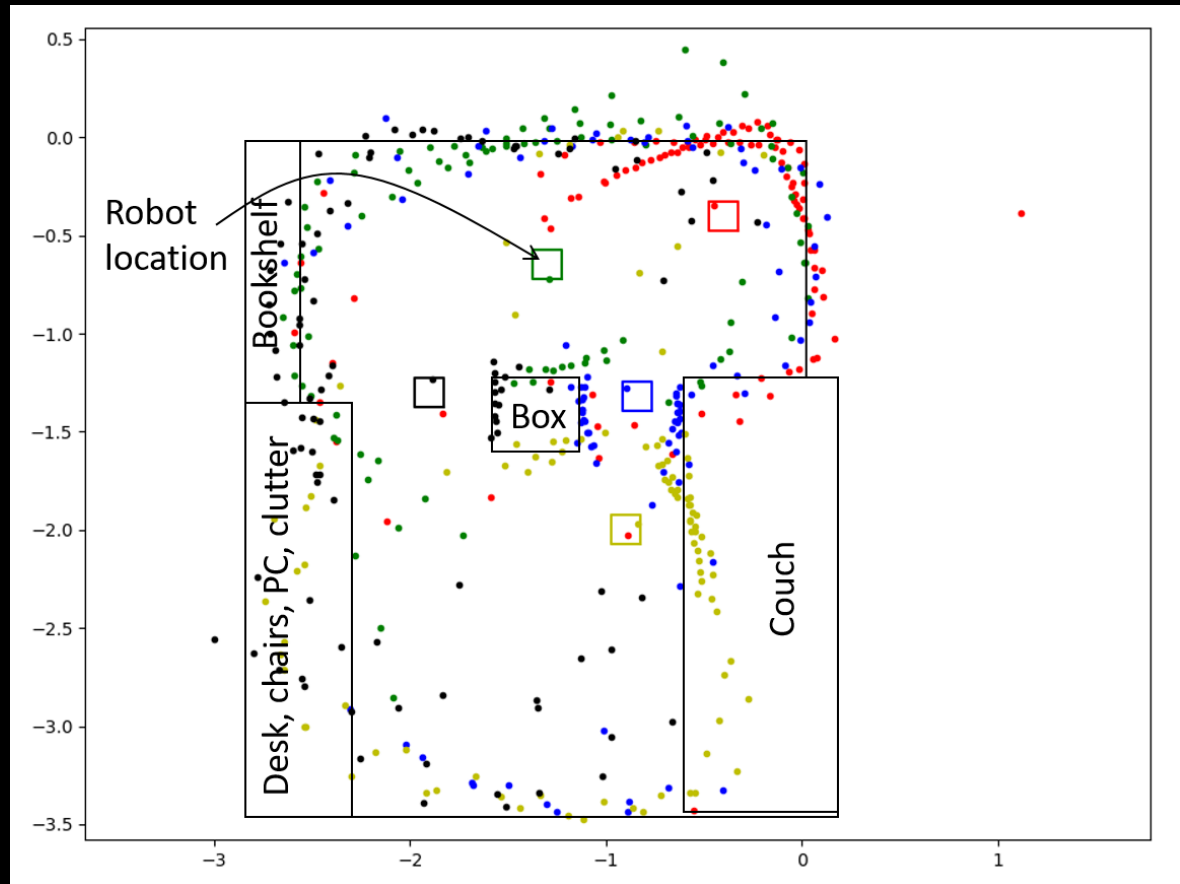
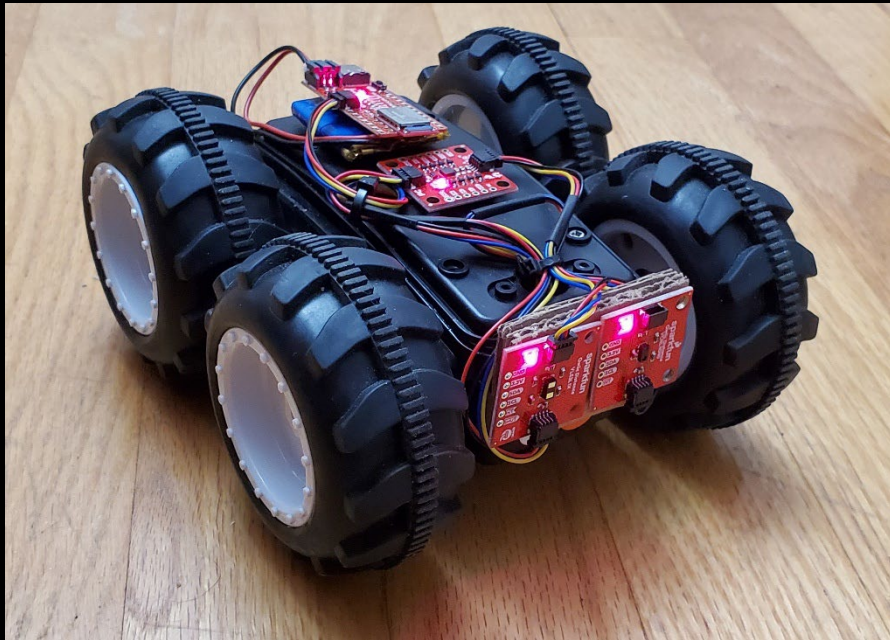
Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

Fast Robots

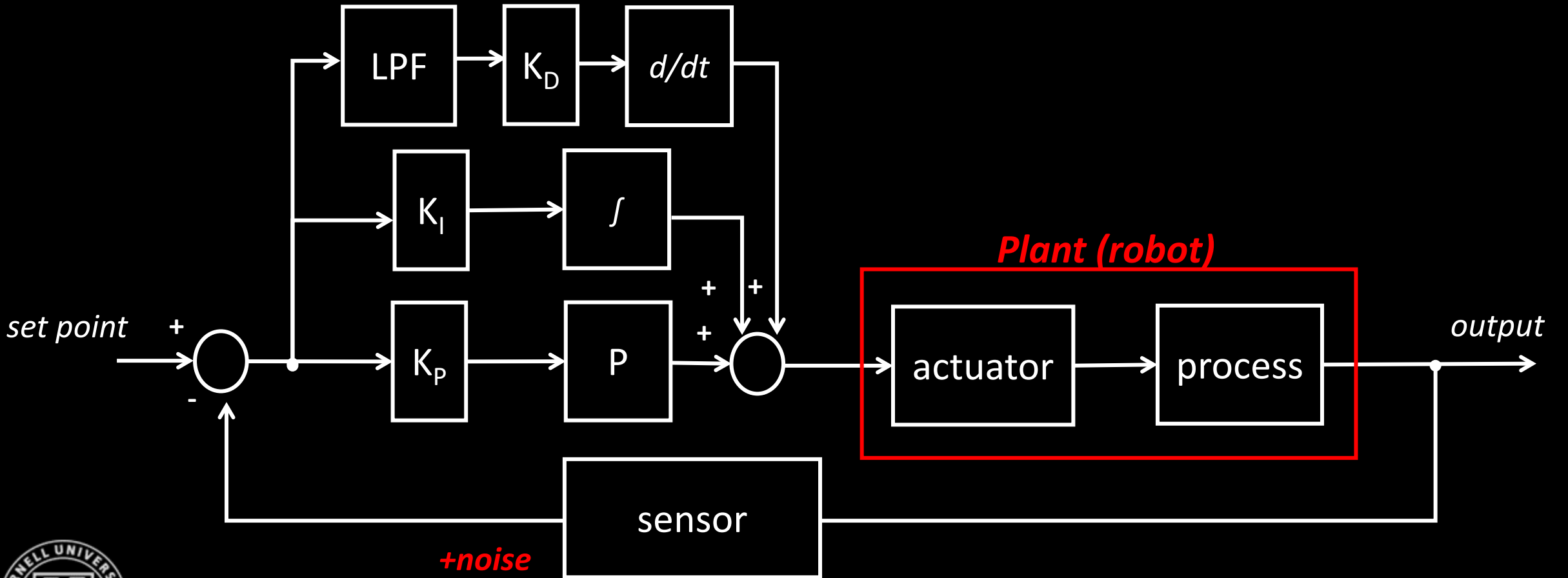
PID (continued)

Feedback Control

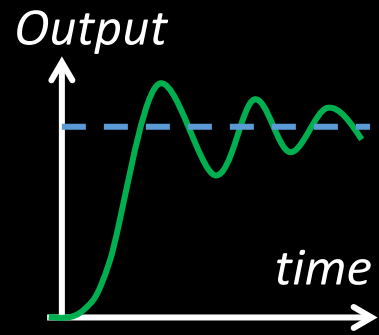
- Use cases in this class
 - Stunt: Maintaining speed at different battery levels and over different surfaces
 - Mapping: Space out sensor readings evenly for mapping
 - Path execution: adhere to generated path plans



PID architecture



PID architecture



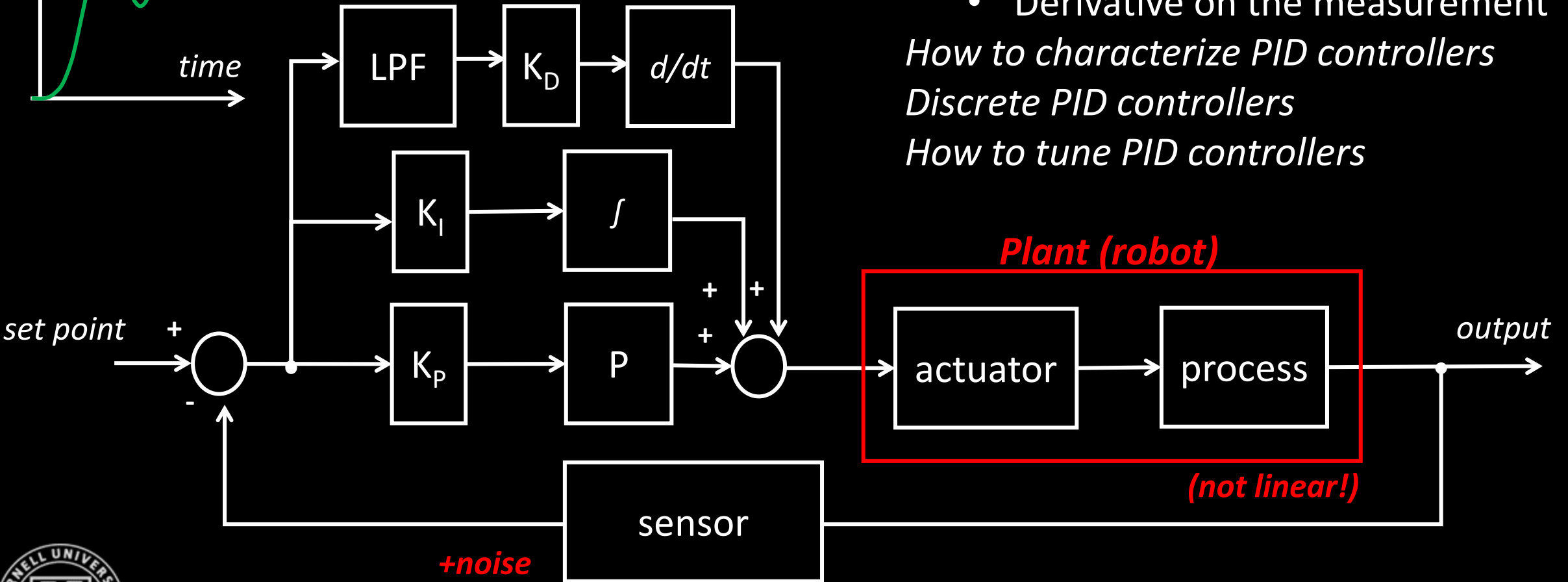
Common things to address:

- Derivative low pass filter
- Integrator wind-up
- Derivative kick
 - Derivative on the measurement

How to characterize PID controllers

Discrete PID controllers

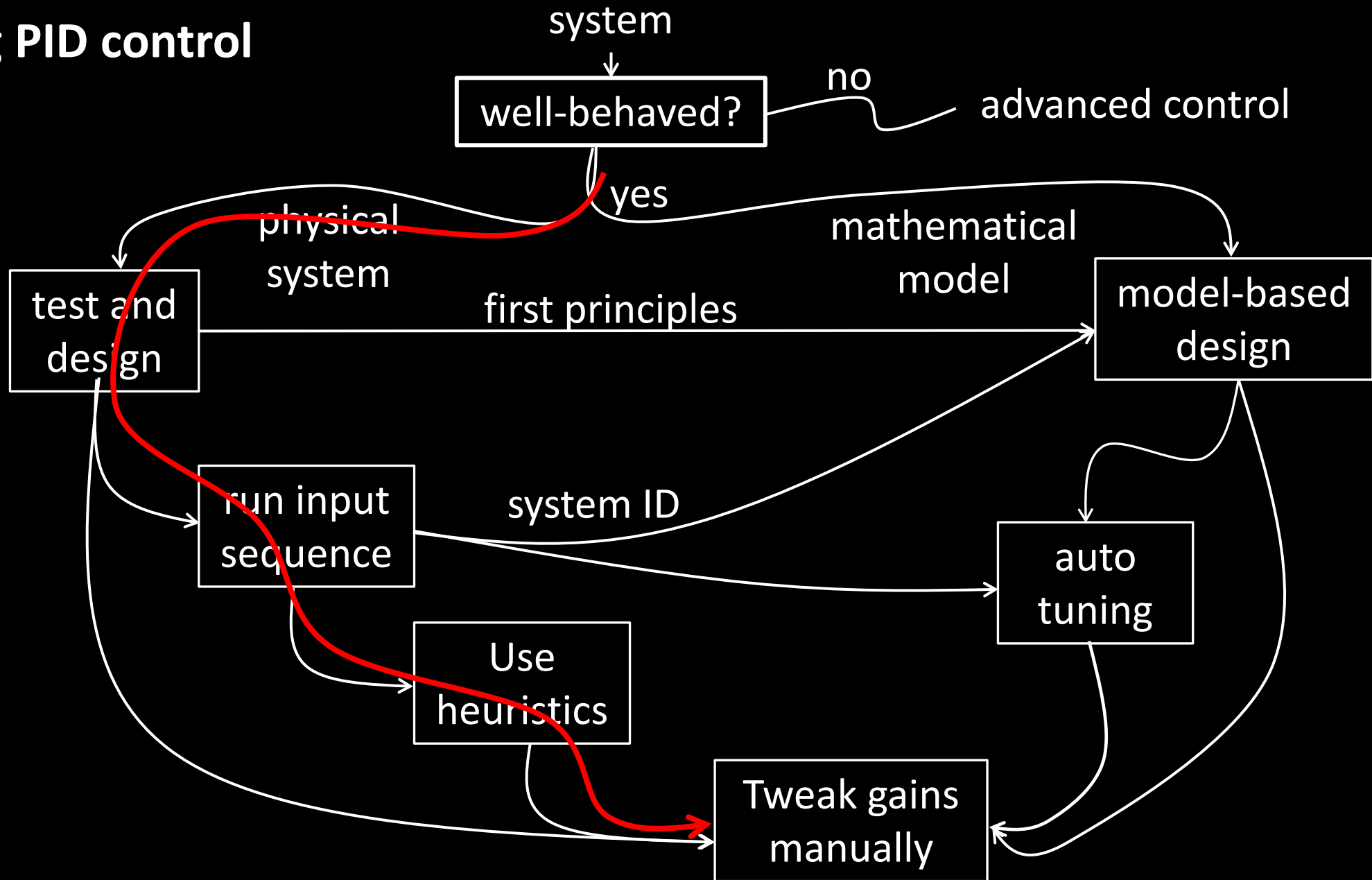
How to tune PID controllers



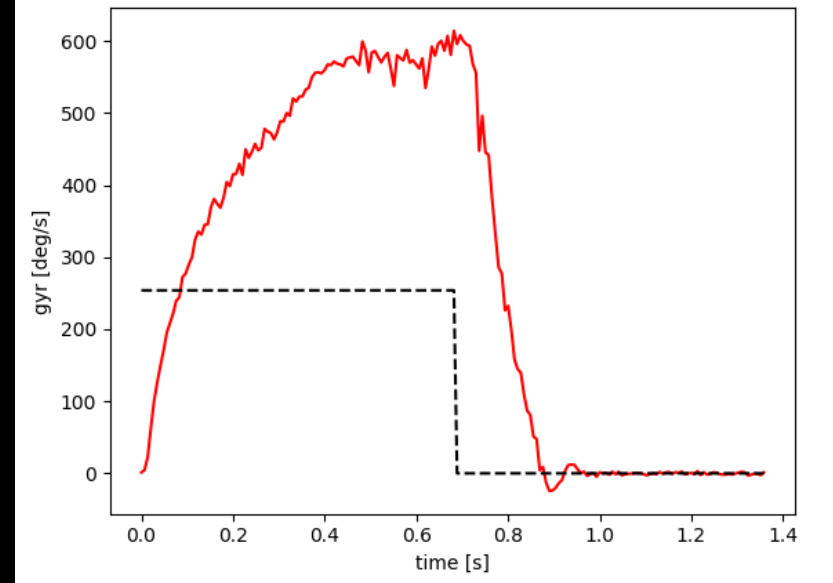
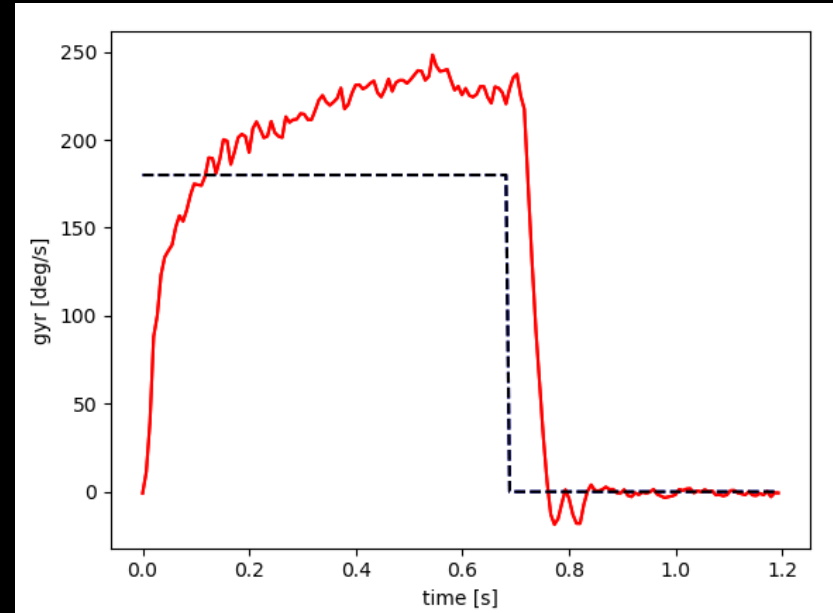
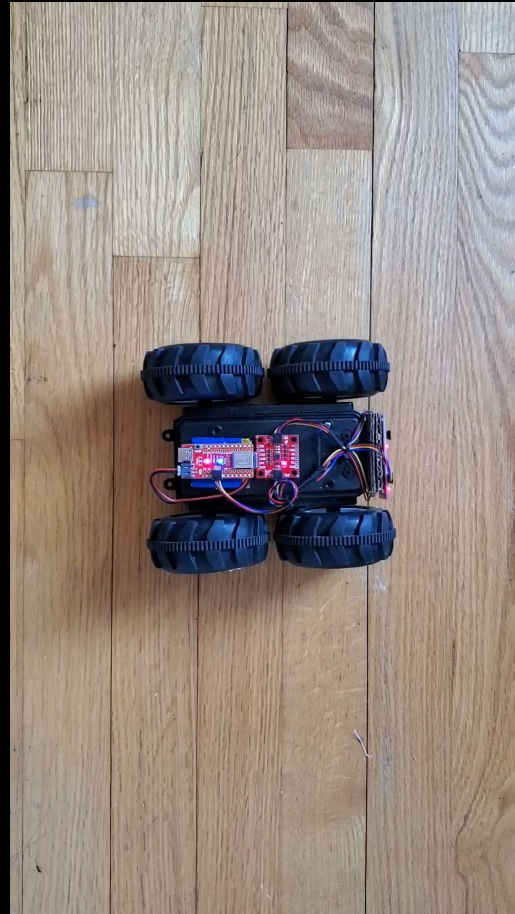
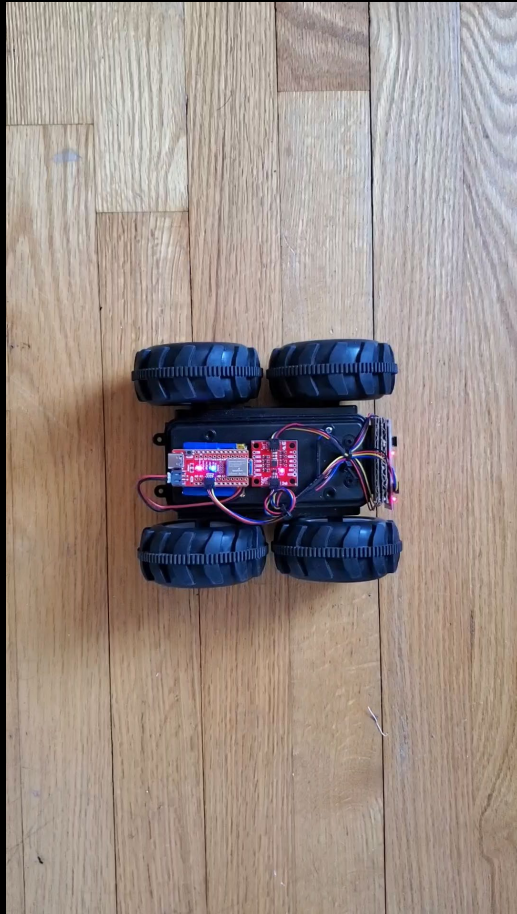
Plant (robot)

(not linear!)

Tuning PID control



Tuning PID control



Tuning PID control (Heuristic)

- Chien, Hornes, and Reswick method

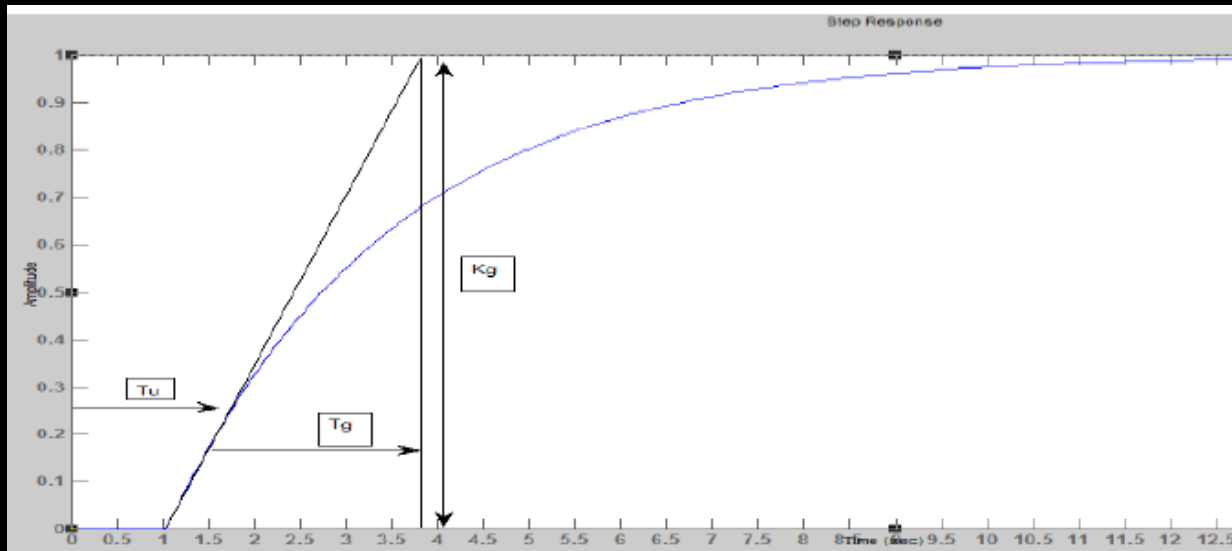
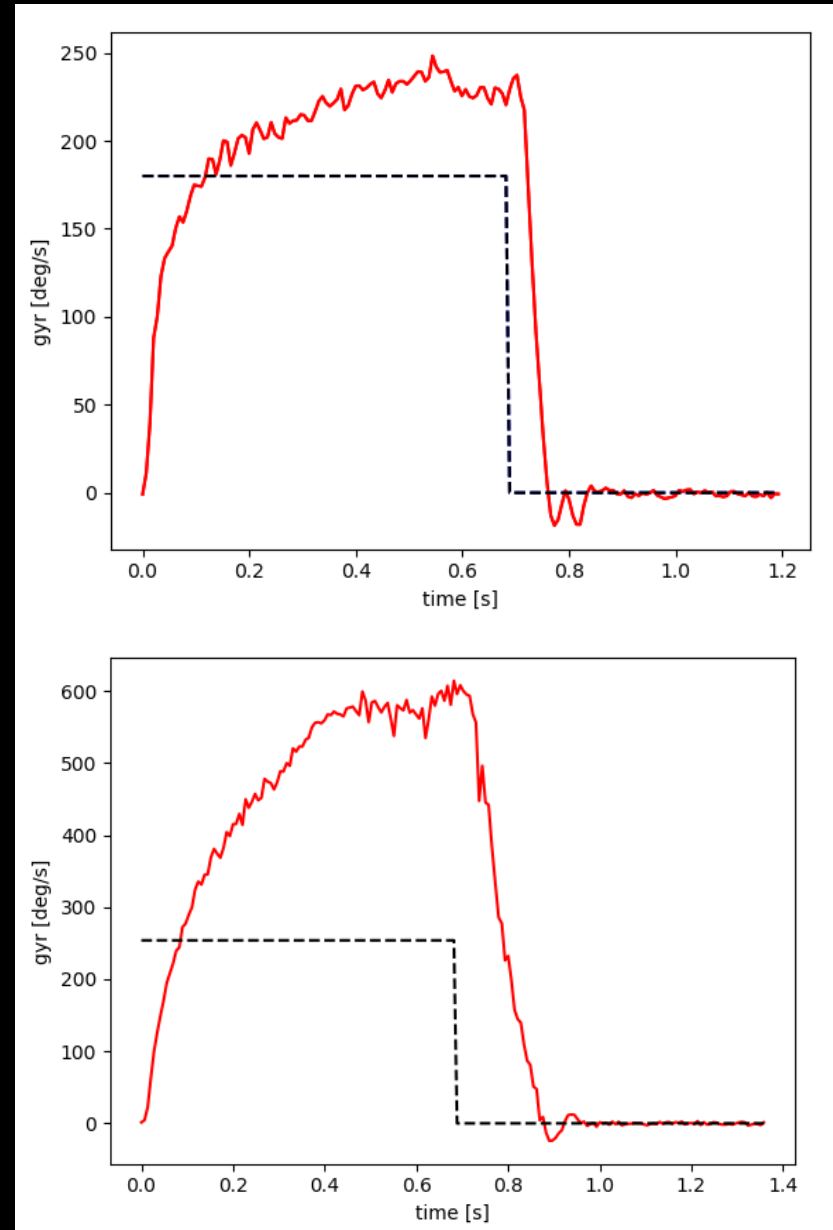


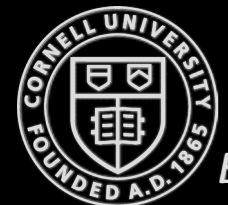
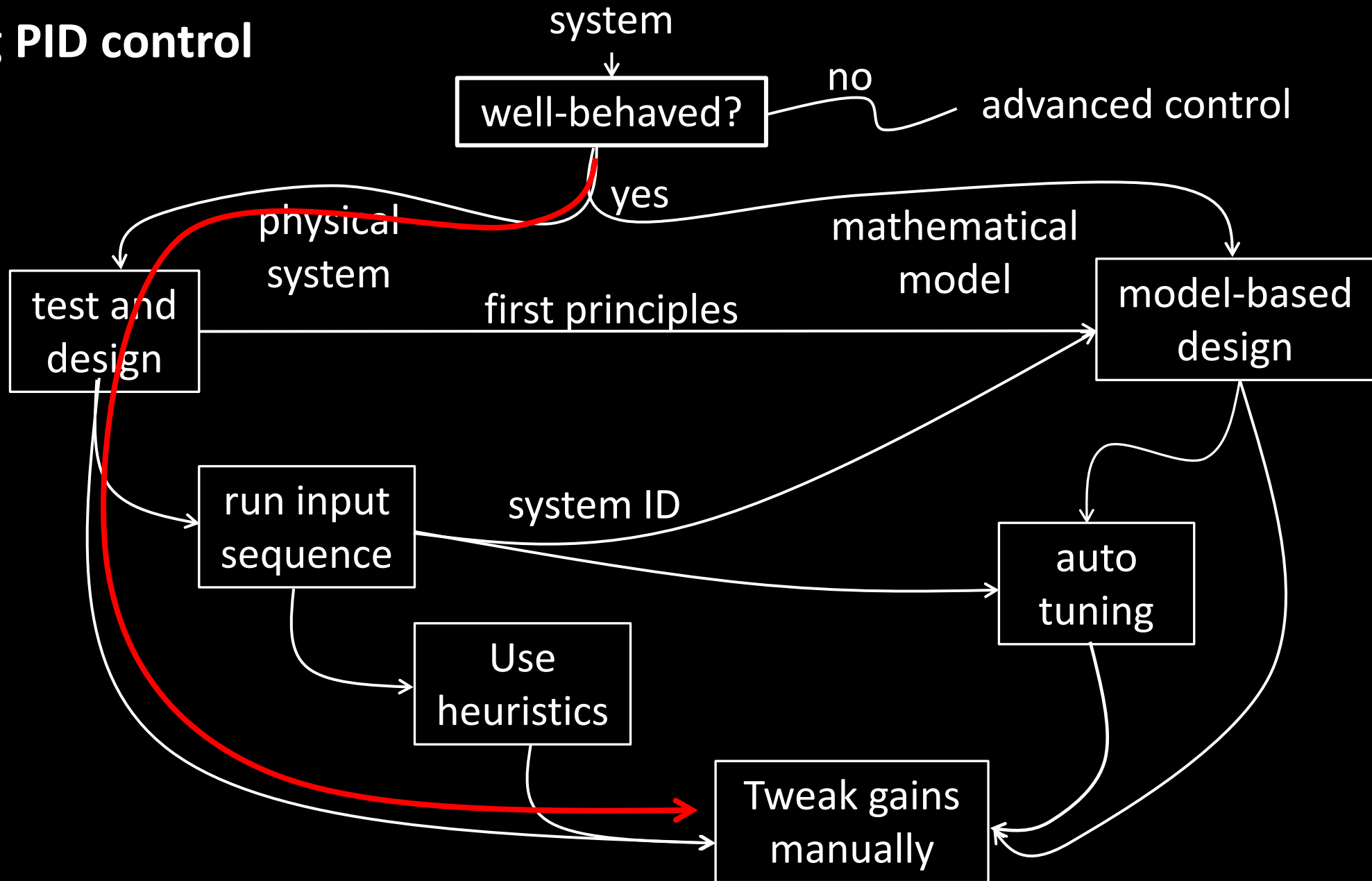
Fig.7. Open loop response of CHR method

Table.11. CHR Compensator

Type of controller	K_p	T_i	T_d
PID	$0.6T_g/T_uK_g$	T_g	$0.5T_u$



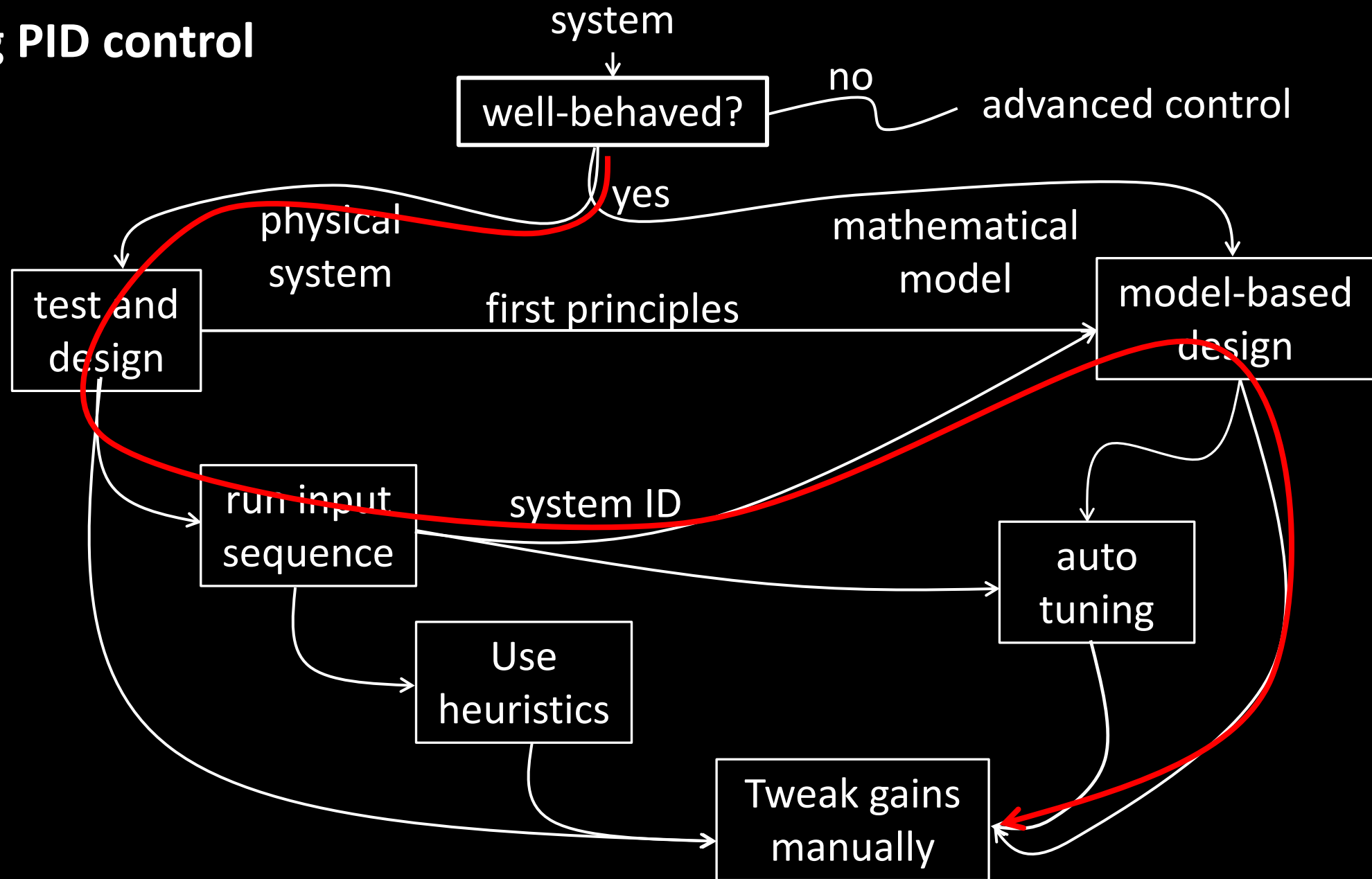
Tuning PID control



PID control (test and design)

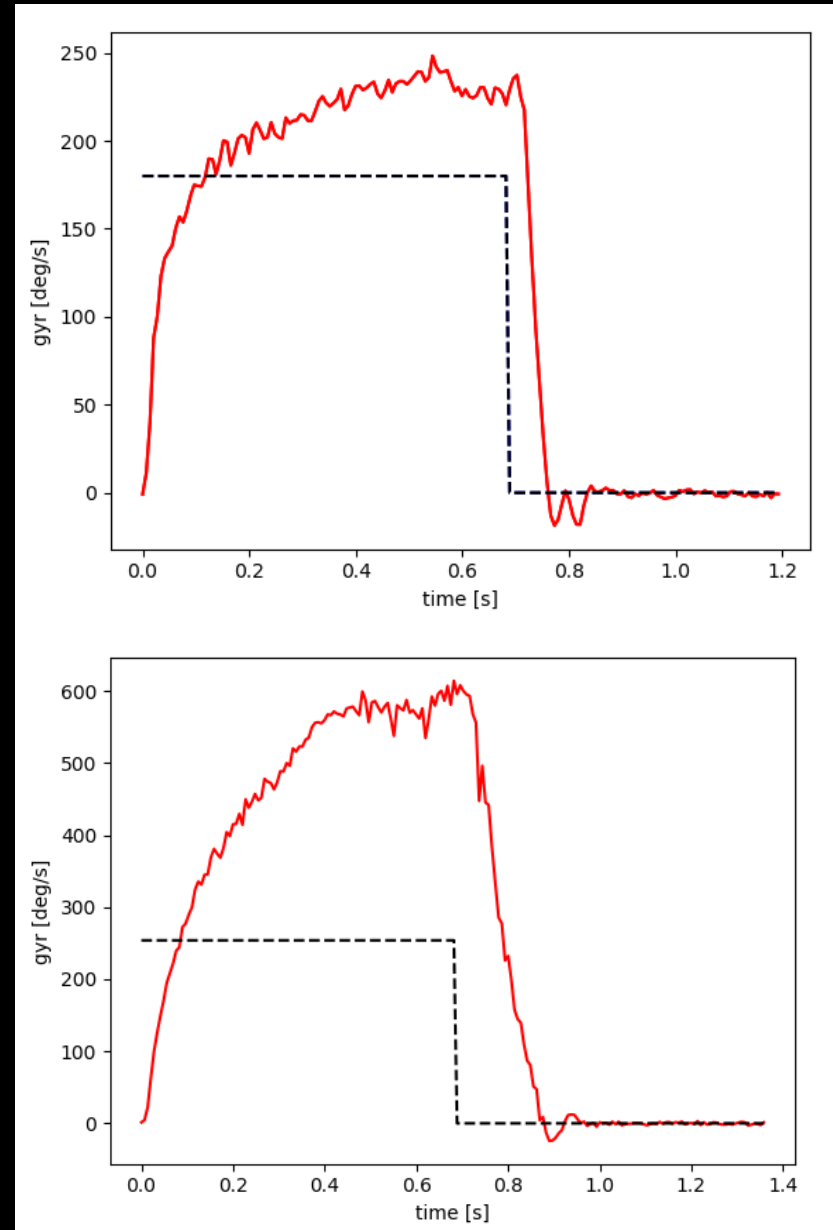
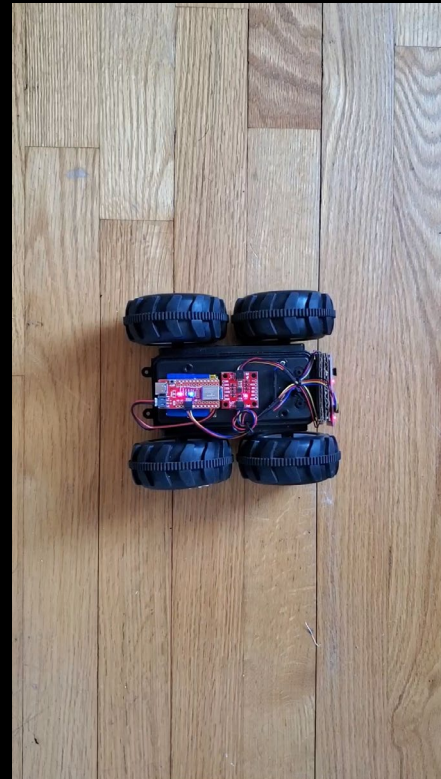
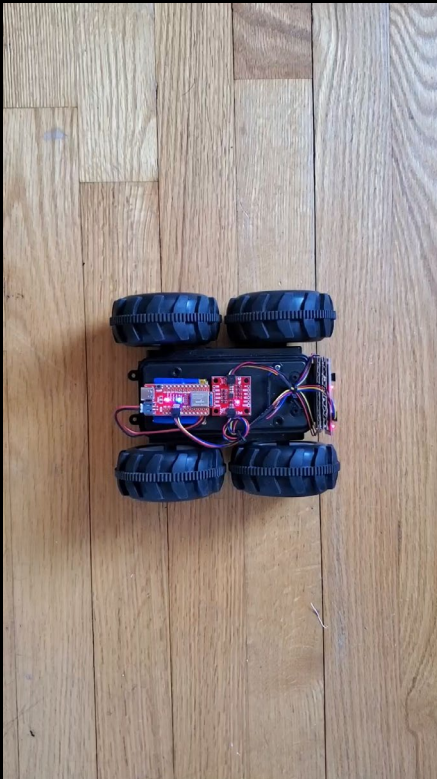
- **Heuristic procedure #1:**
 - Set K_p to small value, K_D and K_I to 0
 - Increase K_D until oscillation, then decrease by factor of 2-4
 - Increase K_P until oscillation or overshoot, decrease by factor of 2-4
 - Increase K_I until oscillation or overshoot
 - Iterate
- **Heuristic procedure #2:**
 - Set K_D and K_I to 0
 - Increase K_P until oscillation, then decrease by factor of 2-4
 - Increase K_I until loss of stability, then back off
 - Increase K_D to increase performance in response to disturbance
 - Iterate

Tuning PID control



Tuning PID control

- Equations of motion
 - First order system...

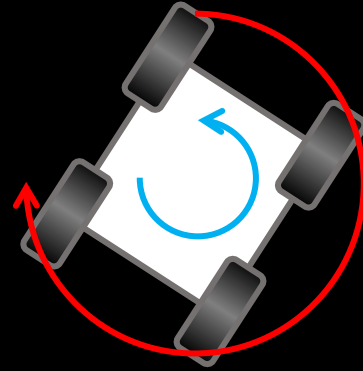


PID control for constant angular speed, $\dot{\theta}$

- Equations of motion

- ~~$x = \dot{\theta}$~~

- $x = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$



$$F = ma$$

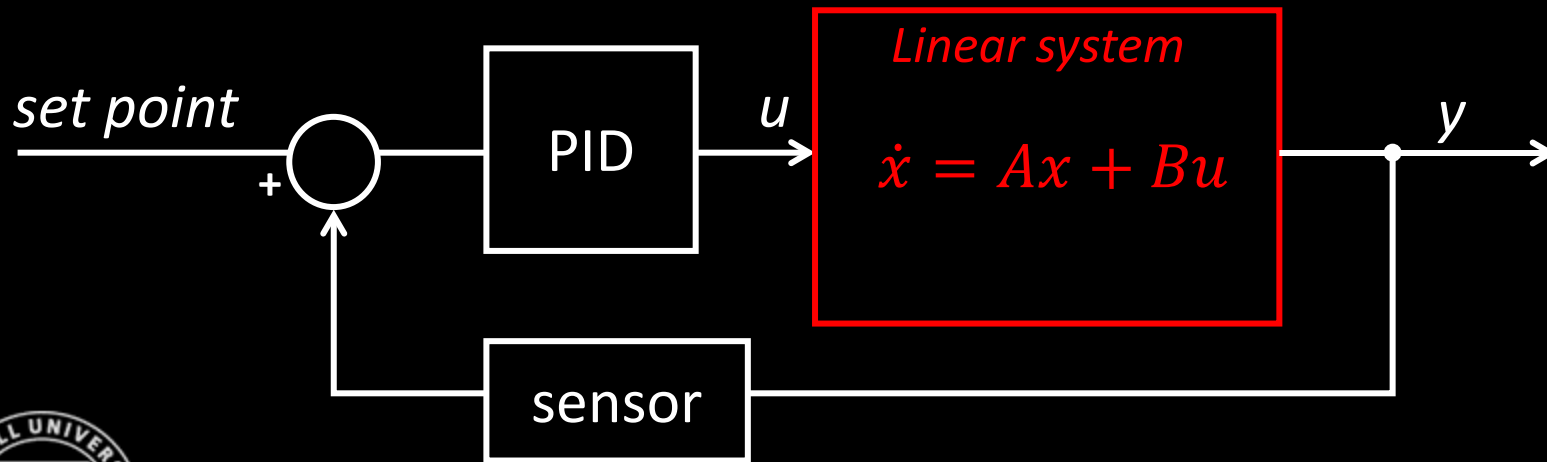
$$\tau = I\alpha$$

$$\tau = I\ddot{\theta}$$

$$u - \dot{\theta}c = I\ddot{\theta}$$

$$\ddot{\theta} = \frac{-\dot{\theta}c}{I} + \frac{1}{I}u$$

~~$$\begin{bmatrix} \ddot{\theta} \end{bmatrix} = \begin{bmatrix} -c \\ I \end{bmatrix} \begin{bmatrix} \dot{\theta} \end{bmatrix} + \begin{bmatrix} 1 \\ I \end{bmatrix} u$$~~



Linear system

$$\dot{x} = Ax + Bu$$

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & \frac{-c}{I} \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{I} \end{bmatrix} u$$

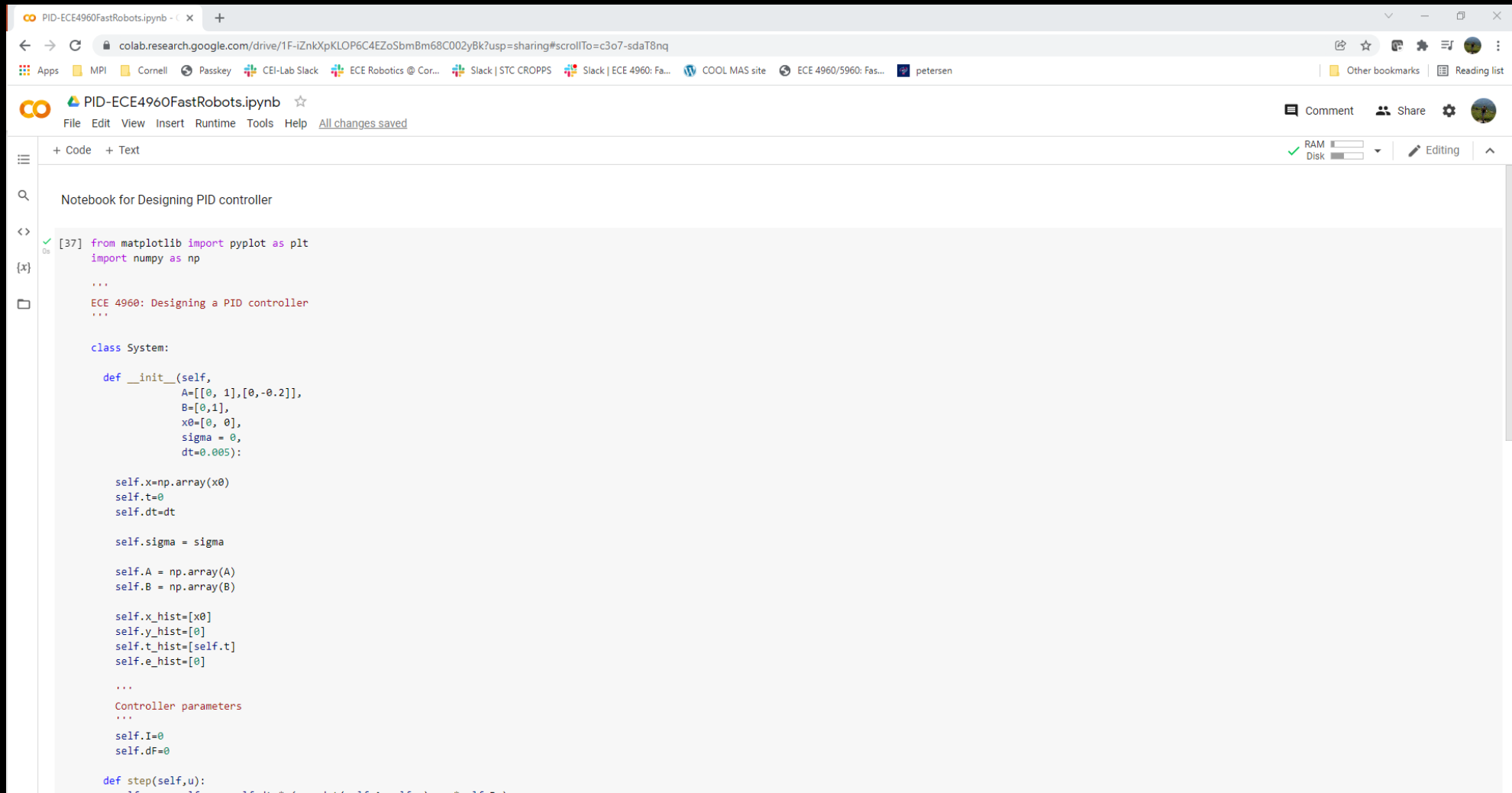
A **B**

Can't affect θ directly



PID control for constant angular speed, $\dot{\theta}$

- <https://bit.ly/3LIAXae>



The screenshot shows a Jupyter Notebook titled "PID-ECE4960FastRobots.ipynb" in a browser window. The notebook content includes the following Python code:

```
[37]: from matplotlib import pyplot as plt
import numpy as np

...

ECE 4960: Designing a PID controller
...

class System:

    def __init__(self,
                 A=[[0, 1],[0,-0.2]],
                 B=[0,1],
                 x0=[0, 0],
                 sigma = 0,
                 dt=0.005):

        self.x=np.array(x0)
        self.t=0
        self.dt=dt

        self.sigma = sigma

        self.A = np.array(A)
        self.B = np.array(B)

        self.x_hist=[x0]
        self.y_hist=[0]
        self.t_hist=[self.t]
        self.e_hist=[0]

        ...

        Controller parameters
        ...

        self.I=0
        self.dF=0

    def step(self,u):
        self.x = self.x + self.dt * ( np.dot(self.A,self.x) + u*self.B )
```

PID control for constant angular speed, $\dot{\theta}$

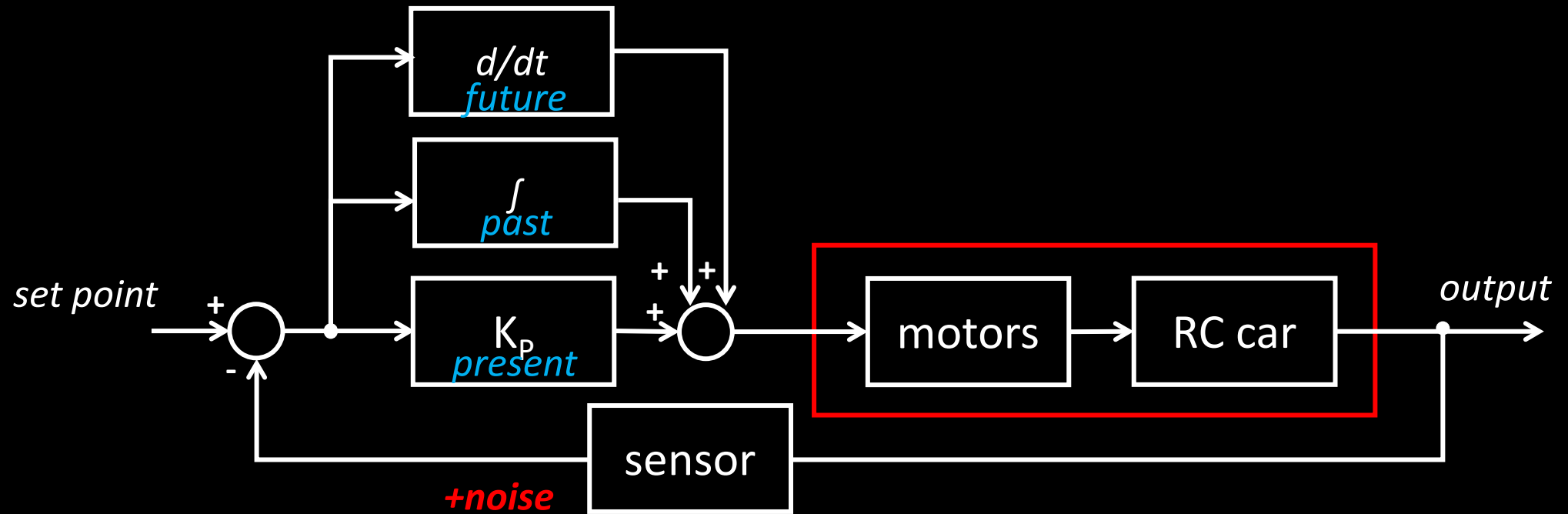
- <https://bit.ly/3LIAXae>

- **Heuristic procedure #1:**
 - Set K_p to small value, K_D and K_I to 0
 - Increase K_D until oscillation, then decrease by factor of 2-4
 - Increase K_P until oscillation or overshoot, decrease by factor of 2-4
 - Increase K_I until oscillation or overshoot
 - Iterate
- **Heuristic procedure #2:**
 - Set K_D and K_I to 0
 - Increase K_P until oscillation, then decrease by factor of 2-4
 - Increase K_I until loss of stability, then back off
 - Increase K_D to increase performance in response to disturbance
 - Iterate

PID control for constant angular speed, $\dot{\theta}$

- <https://bit.ly/3LIAXae>
- Overshoot ($K_p = 10, K_i = 100$)
- Dampening ($K_p = 10, K_i = 100, K_D = 0.8$)
- Noise ($\sigma = 0.1$)
- LPF ($\alpha = 0.05$)
- Derivative kick ($\alpha = 1, \sigma = 0$)

PID control of a 2nd order system



$$1^{\text{st}} \text{ order system: } \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & \frac{-c}{I} \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{I} \end{bmatrix} u$$

$$2^{\text{nd}} \text{ order system: } \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ cst & \frac{-c}{I} \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{I} \end{bmatrix} u$$

Looking ahead: lab 6, PID control

- Control for fast mode
 - Stunt: orientation
 - Stunt: speed control
- Control for slow mode
 - Mapping: angular speed
 - Path execution: position control

Biggest limitation?

- Sensor noise
- Sensor sampling time
- PID control is preferably 5-10 times faster than your system
- *Lab 7 Kalman filter*
- *Lab 8 Stunt*

Next three lectures

- Control theory
 - Linear systems
 - Eigenvectors
 - Stability
 - Controllability
 - Observability
 - Kalman filters

$$\dot{x} = Ax + Bu$$

These should look familiar from..

- MATH 2940 Linear Algebra
- ECE3250 Signals and systems
- ECE5210 Theory of linear systems
- MAE3260 System Dynamics
- etc...