Control Overview

Most generally, the goal of any controller is to make a machine do what we want it to

Q: What's the simplest controller you can think of?

1) no control ⟶ just turn on and go (no correction for errors, etc.)

2) on/off

- thermostats
- air compressors

pseudo-code for Air Conditioner

```
if (temp > desired temp){
   turn AC on;
else {
   turn AC off;
}
```

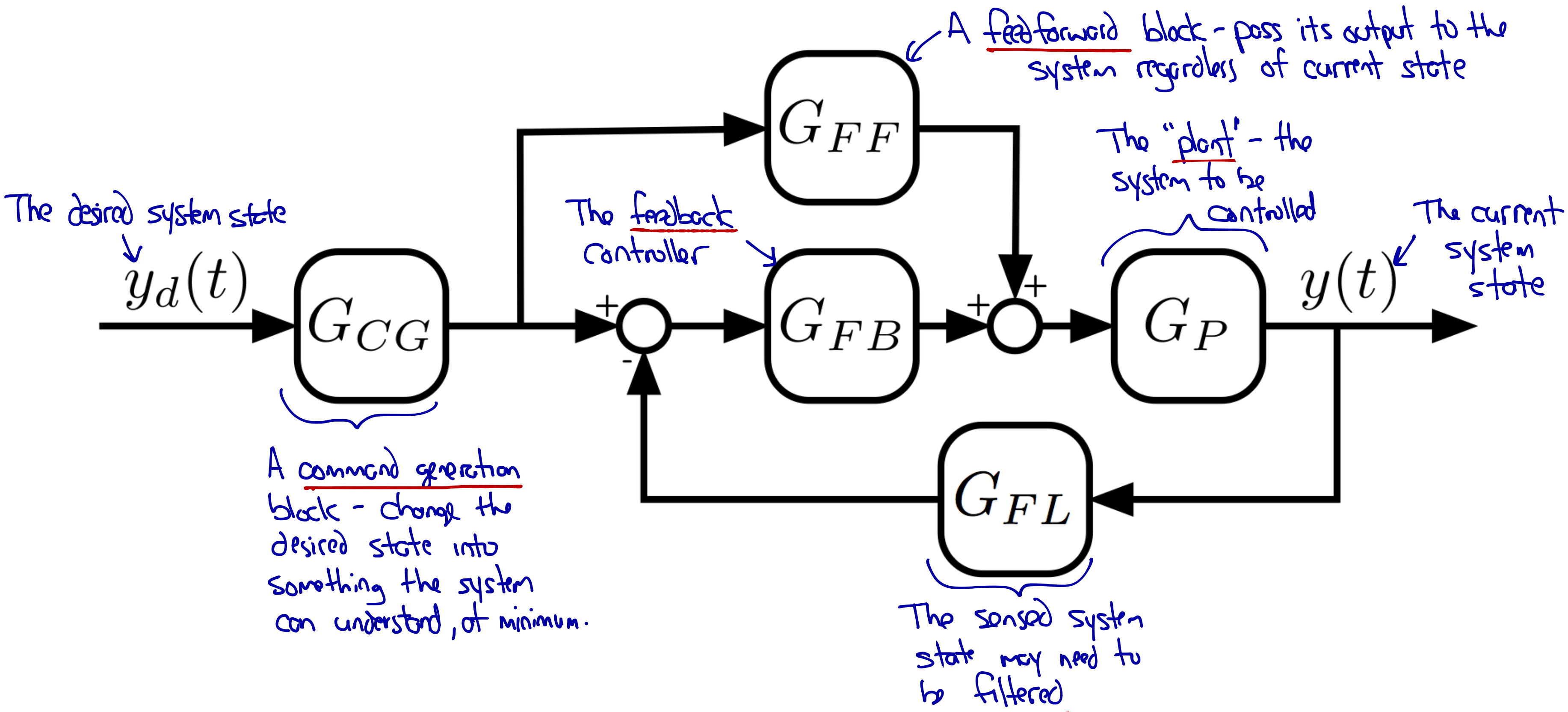
Q: What are the problems with this?

- no accounting for magnitude of error } make control input proportional to error
- doesn't consider rate of change of error } make control input prop to $\frac{d}{dt}(\text{error})$
- can result in rapid switches
- will (almost) always deviate from desired } make control input prop to $\int(\text{error})dt$

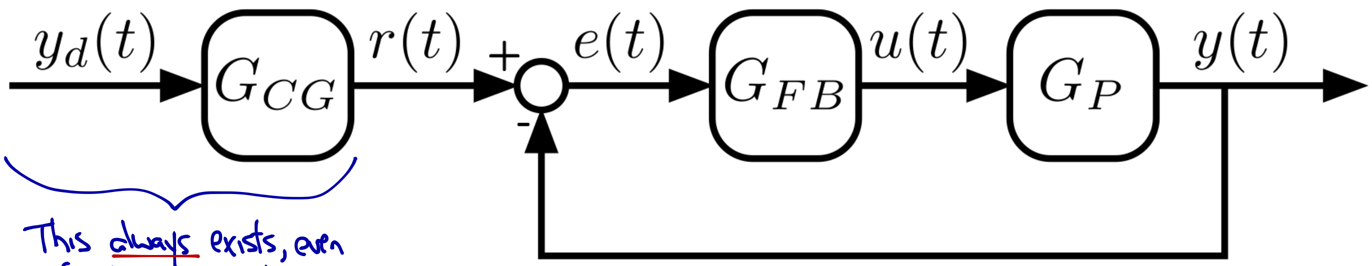
PID Feedback Control

Block diagrams: represent the elements of the system as blocks

The input/output relationships of the blocks are shown as lines



Basic Feedback Control



This always exists, even if not explicitly shown.

$r(t)$ does not have to be $y_d(t)$

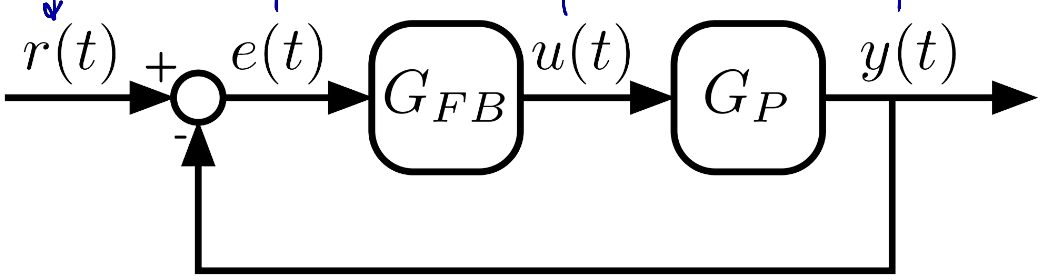
The "textbook" version

The reference command

$e(t) = r(t) - y(t)$

$u(t) = G_{FB}\, e(t)$

$y(t) = G_P\, u(t)$
current state



G_{FB}: Feedback controller
acts on $e(t)$, the error between reference and current state

G_P: The "plant" - the system to be controlled

Q: What should G_{FB} be?

· proportional to error (if I'm far away from where I want to be, a larger corrective action is needed)

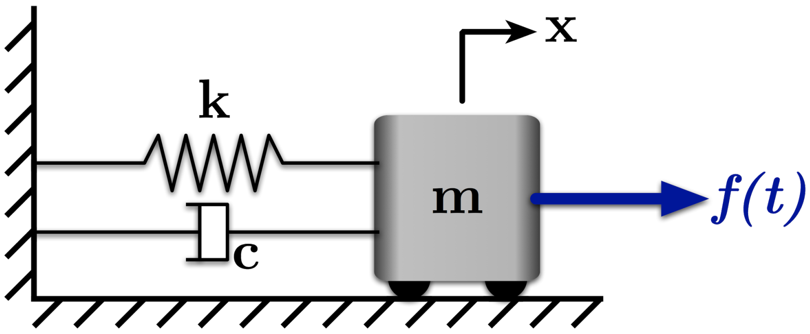
Proportional Control

Proportional Control

$$u(t) = k_P\, e(t) = k_P\left(x_d(t) - x(t)\right)$$

Proportional Control Example

$$u(t) = k_p e(t) = k_p\left(x_d(t) - x(t)\right)$$



$$m\ddot{x} + c\dot{x} + kx = f$$

taking the Laplace Transform

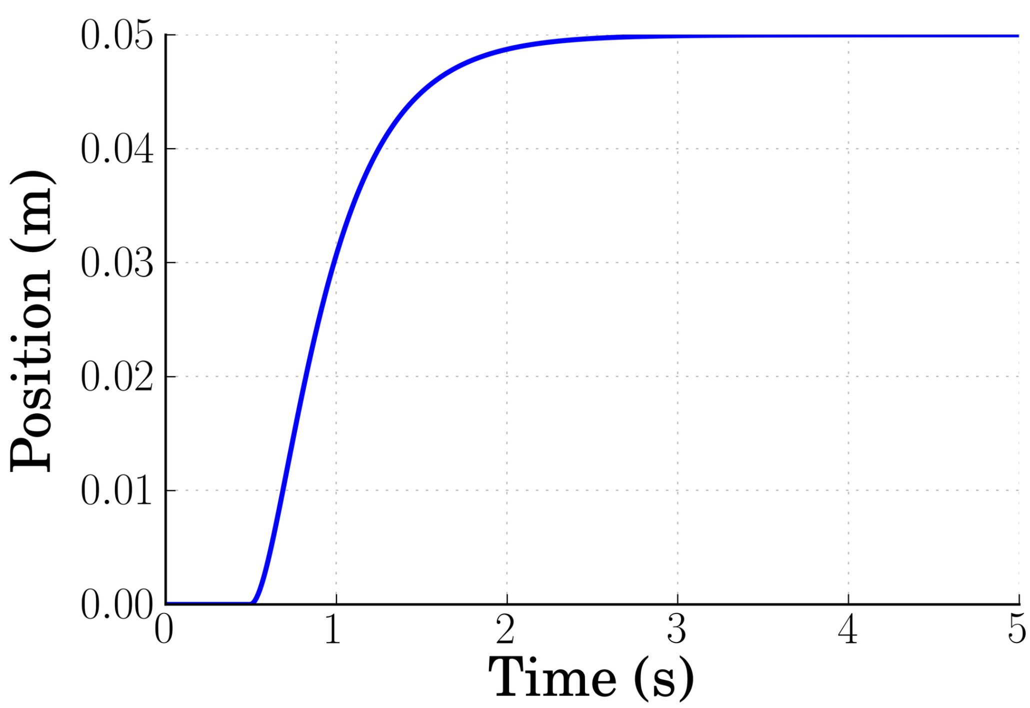
$$(ms^2 + cs + k)X(s) = F(s)$$

$$\frac{X(s)}{F(s)} = \frac{1}{ms^2 + cs + k} \quad \Big\{ \text{Transfer function}$$

Let $m = 1\,kg$, $k = 20\,N/m$, $c = 10\,N/m^2$

Goal: A unit change in position.

Q: What will the response to a unit step in $f(t)$ look like?



This is the open-loop response

Open-loop - input is not adjusted according to output

Q: Problems with this response?

- steady-state error
- slow rise time

Try proportional control:

Let $f(t) = k_p\left(x_d(t) - x(t)\right) \longrightarrow m\ddot{x} + c\dot{x} + kx = k_p(x_d - x)$

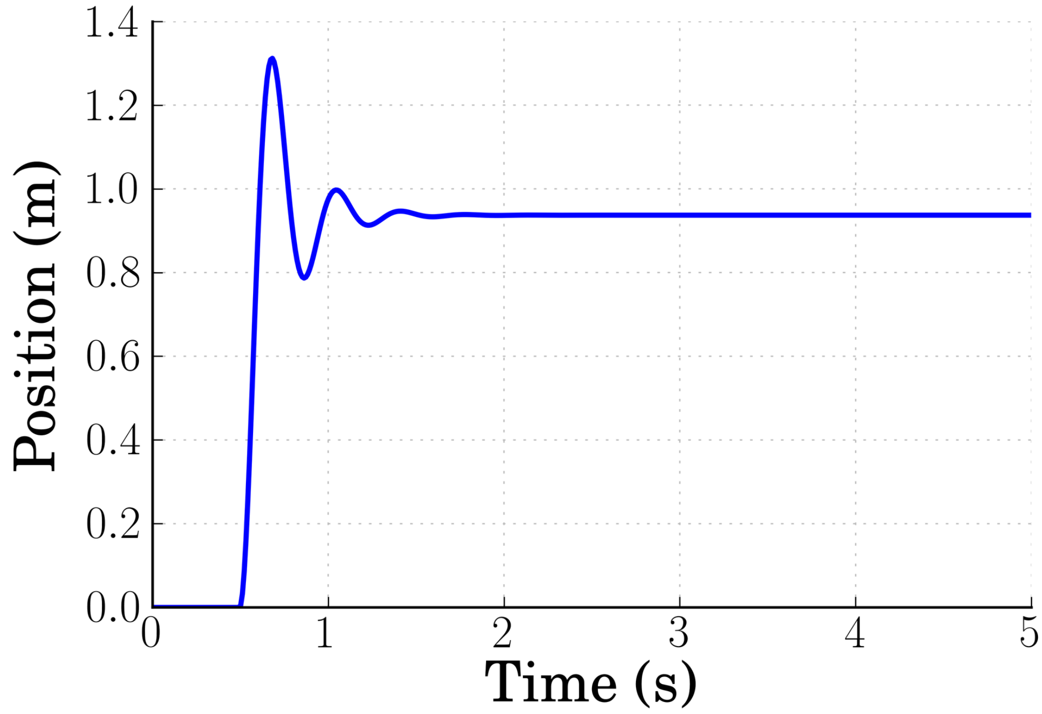
$$m\ddot{x} + c\dot{x} + (k + k_p)x = k_p x_d$$

$$\left[ms^2 + cs + (k + k_p)\right]X(s) = k_p X_d(s)$$

$$\frac{X(s)}{X_d(s)} = \frac{k_p}{ms^2 + cs + (k + k_p)} \quad \Big\} \text{closed-loop transfer function}$$

Proportional Control Example (cont.)

Try $k_p = 300$



Q: Better?

- shorter rise time
- less steady-state error

Q: Worse?

- Introduced overshoot
- Introduced oscillation

Q: How can we improve?

- better selection of k_p (maybe)
- add derivative gain

Proportional-Derivative Control

$$f(t) = k_p e(t) + k_d \dot{e}(t) = k_p(x_d - x) + k_d(\dot{x}_d - \dot{x})$$

sub. into Eq of Motion:

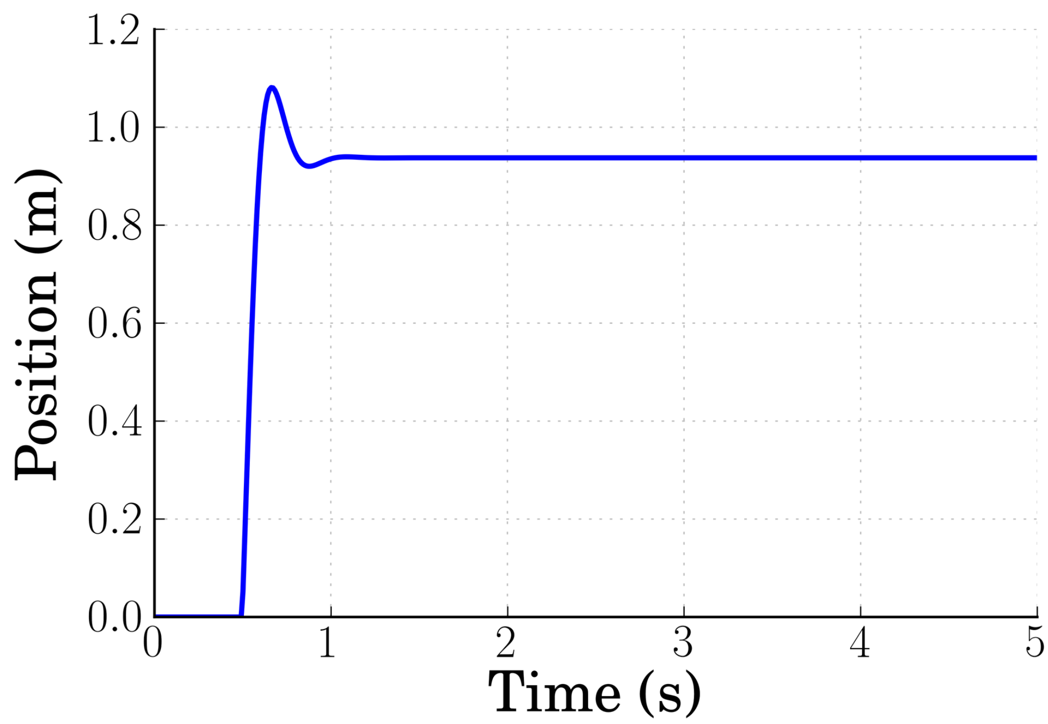
$$m\ddot{x} + c\dot{x} + kx = k_p(x_d - x) + k_d(\dot{x}_d - \dot{x}) \longrightarrow m\ddot{x} + (c + k_d)\dot{x} + (k + k_p)x = k_p x_d + k_d \dot{x}_d$$

Take the Laplace Transform

$$\left[ms^2 + (c+k_d)s + (k+k_p)\right] X(s) = \left[k_d s + k_p\right] X_d(s) \longrightarrow \frac{X(s)}{X_d(s)} = \frac{k_d s + k_p}{ms^2 + (c+k_d)s + (k+k_p)}$$

} closed-loop transfer func

Let $k_p = 300$ and $k_d = 10$



Q: Better?

- less overshoot
- less oscillation (better settling time)

Q: Problems?

- still steady-state error
- need to know velocity to implement

not trivial!

Q: How can we eliminate the steady-state error?

Q: Why is there steady-state error?

force generated by the controller not enough to overcome other system forces

For this example

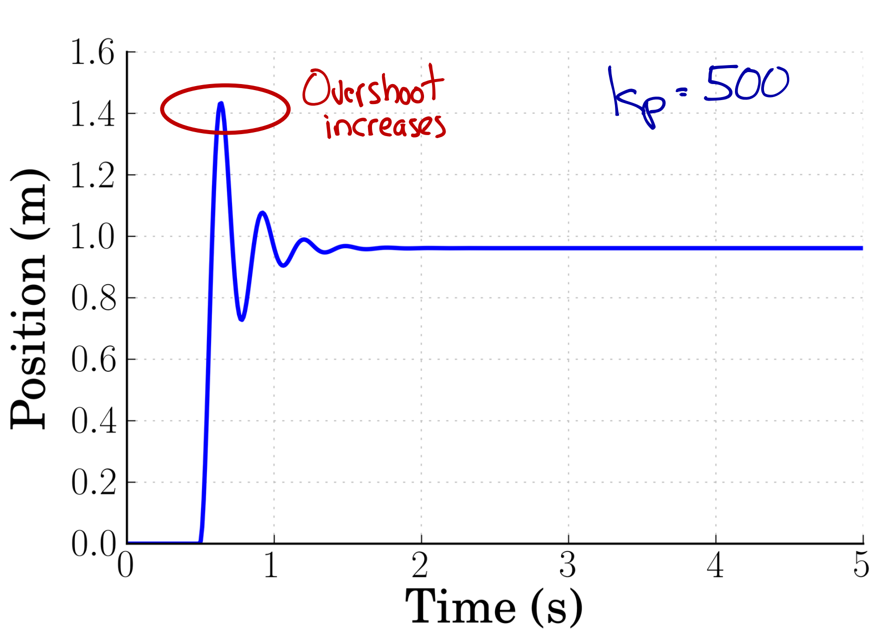
$$\cancel{m\ddot{x}} + \cancel{c\dot{x}} + kx = k_p(x_d - x) + \cancel{k_d(\dot{x}_d - \dot{x})}$$

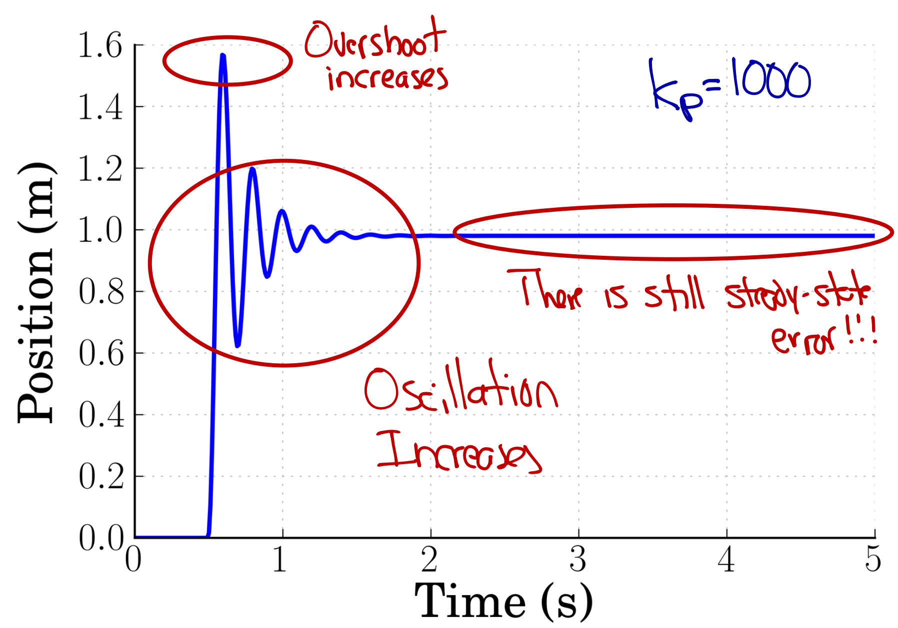
– at steady-state, all "motion" terms are zero

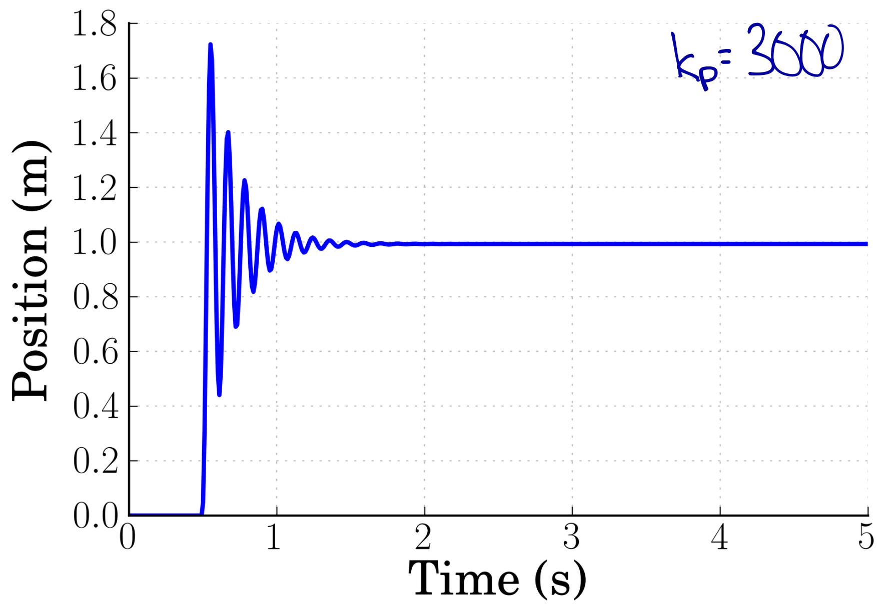
$$kx = k_p(x_d - x)$$

at x_d, $F_{sp} = kx_d$ ← Our controller doesn't generate enough force to balance this.

Q: Why not just increase k_p?







Q: What other problem is not shown in these plots?

- actuator effort (the force the controller is generating)
  - higher peak
  - oscillatory

Proportional-Integral-Derivative (PID) Control

Add an integral controller to remove steady-state error

Q: Integral Control?

"goal" - small errors sum over time to create a large-enough value to induce corrective action

$$f(t) = k_p e + k_d \dot{e} + k_i\left[\int e(t)dt\right] = k_p(x_d - x) + k_d(\dot{x}_d - \dot{x}) + k_i\left[\int (x_d - x)dt\right]$$

Sub. into Eq of Motion:

$$m\ddot{x} + c\dot{x} + kx = k_p(x_d - x) + k_d(\dot{x}_d - \dot{x}) + k_i\left[\int (x_d - x)dt\right]$$

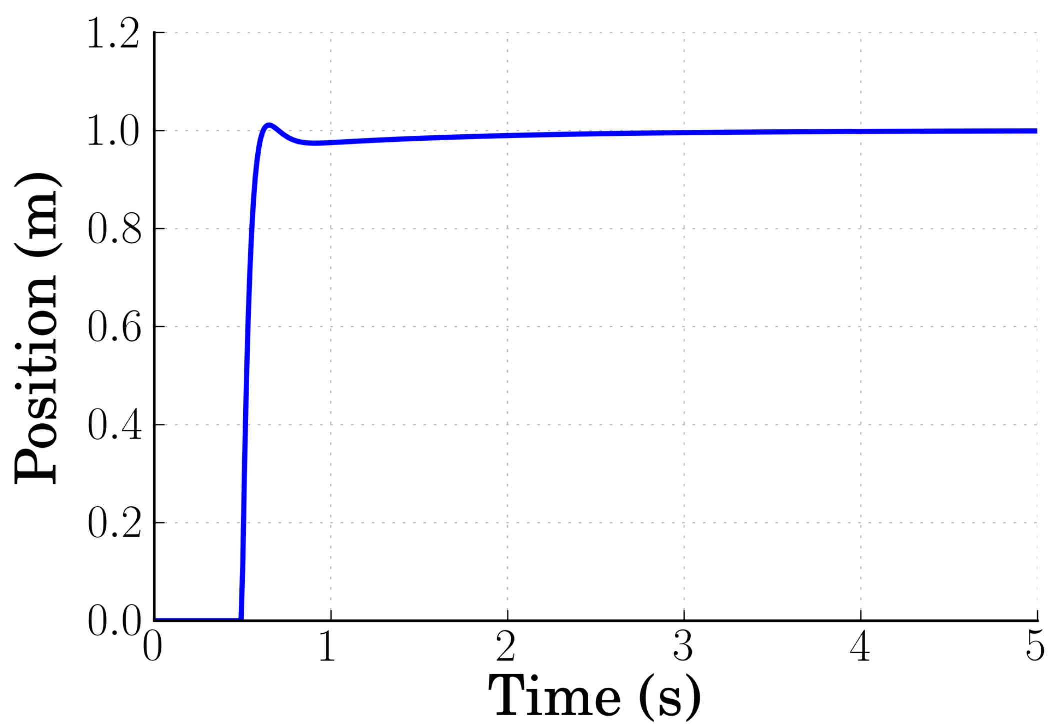
$$m\ddot{x} + (c + k_d)\dot{x} + (k + k_p)x + k_i\int x\,dt = k_p x_d + k_d \dot{x}_d + k_i \int x_d\,dt$$

Take the Laplace Transform:

$$\left[ms^2 + (c + k_d)s + (k + k_p) + \frac{1}{s}k_i\right]X(s) = \left[k_d s + k_p + \frac{1}{s}k_i\right]X_d(s) \leftarrow \text{multiply by } s$$

$$\left[ms^3 + (c + k_d)s^2 + (k + k_p)s + k_i\right]X(s) = \left[k_d s^2 + k_p s + k_i\right]X_d(s)$$

$$\frac{X(s)}{X_d(s)} = \frac{k_d s^2 + k_p s + k_i}{ms^3 + (c + k_d)s^2 + (k + k_p)s + k_i} \quad \Big\} \text{ closed loop transfer function}$$



$k_p = 350$

$k_d = 25$

$k_i = 300$

PID Trends

As this gain is increased ⟶ These general trends occur.

Gain	Rise Time	Overshoot	Settling Time	Steady-State Error
k_p	$\downarrow$	$\uparrow$	$\approx$	$\downarrow$
k_d	$\approx$	$\downarrow$	$\downarrow$	$=$
k_i	$\downarrow$	$\uparrow$	$\uparrow$	$\rightarrow 0$

Questions to Answer & Things to Consider

1) How do we measure the states we need for feedback?

   NOT TRIVIAL!!!

2) What about sensor noise?

3) How do we choose the gains (k_p, k_d, k_i)?
   - Optimization
   - Heuristic Tuning Methods/algorithms ⟵ Ziegler-Nichols Method
   - Iterative/Brute Force Methods ⟵ "Twiddle"
   - Trial-and-Error

4) How do we actually implement?