Lecture 13

Cont. Linear Control of Manipulators and PID Controller and Tuning

Alexandros Lioulemes, PhD

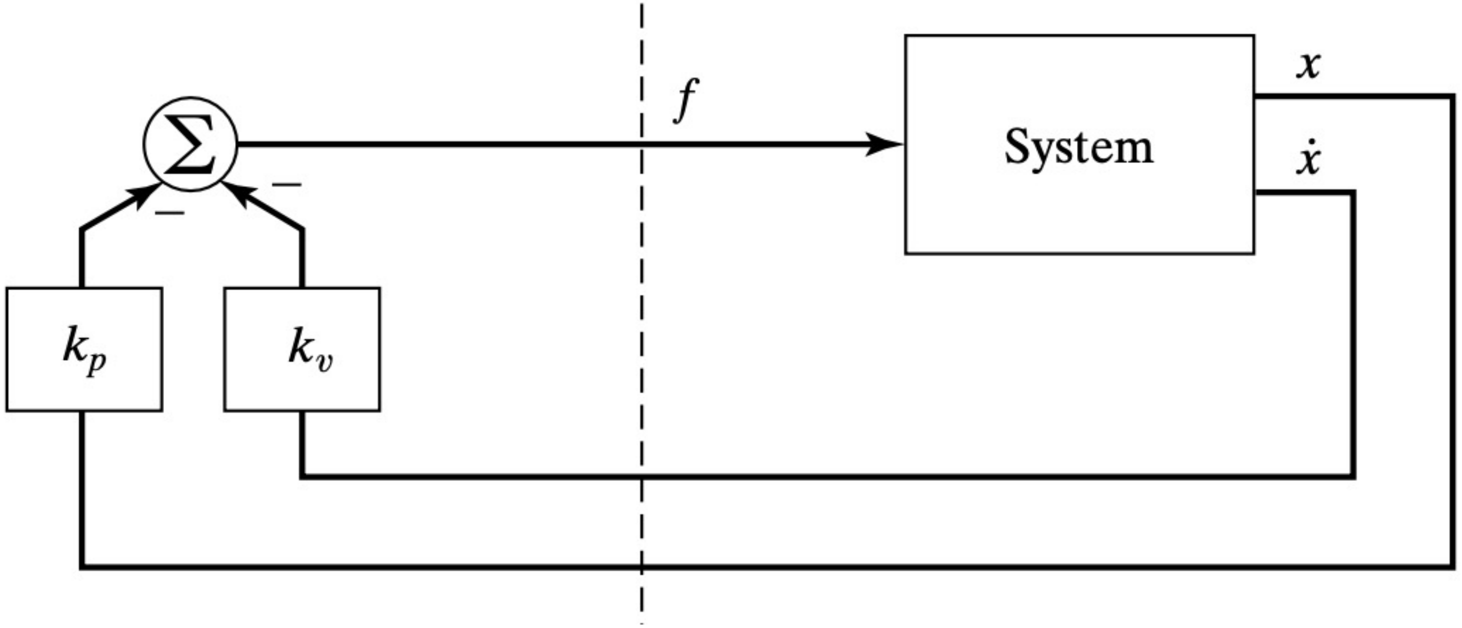
Control of Second-Order Systems

- **Position-regulation system:** It simply attempts to maintain the position of the block in one fixed place regardless of disturbance forces applied in the block.

$$m\ddot{x} + b'\dot{x} + k'x = 0,$$

$$b' = b + k_v$$

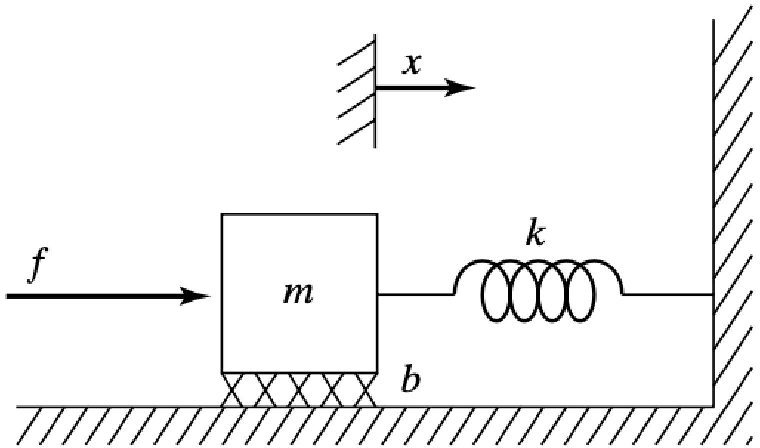
$$k' = k + k_p.$$

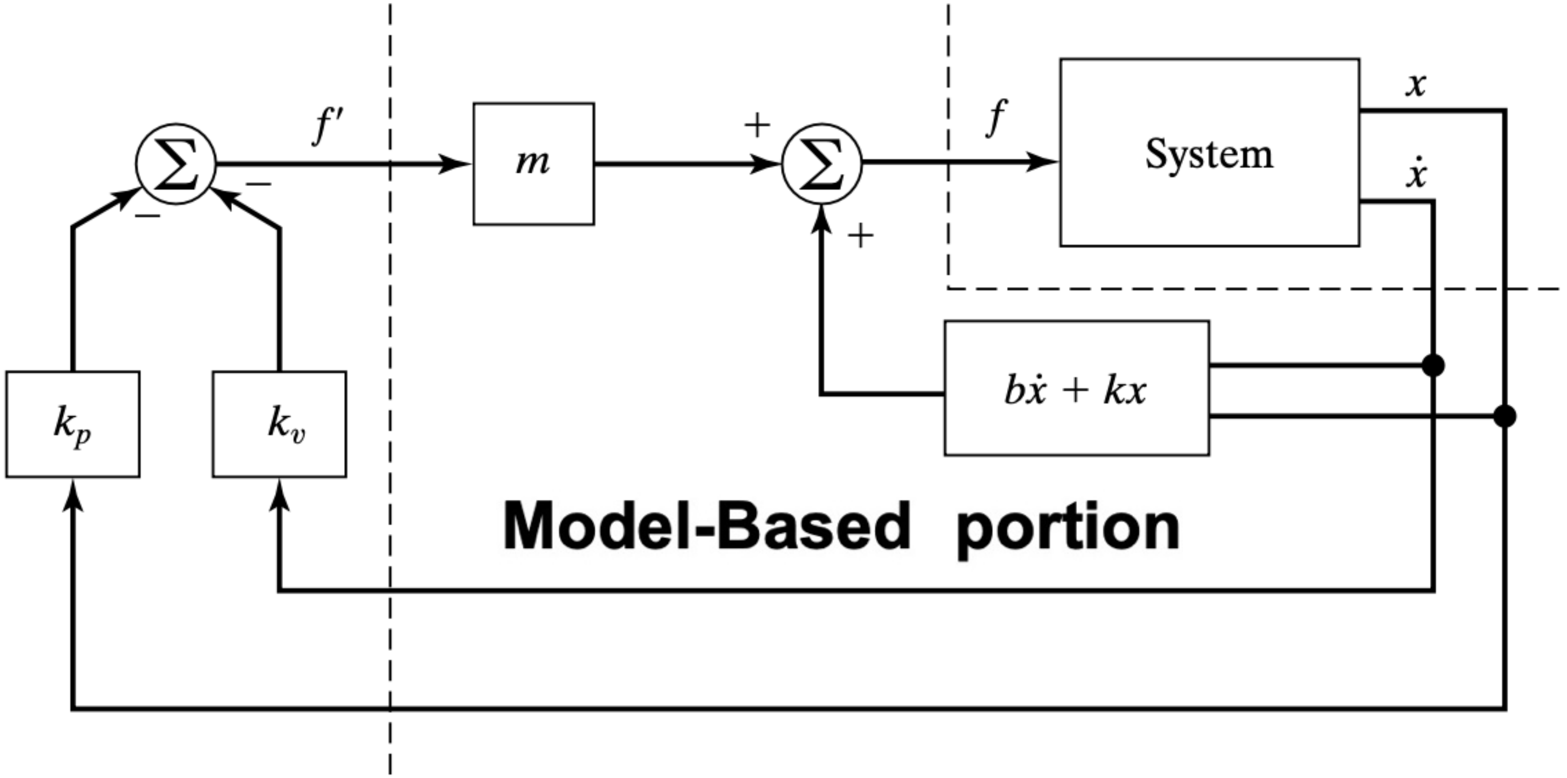


Control-Law Partitioning

- **Model-Based portion:** $f = \alpha f' + \beta$, where α, β are functions or constants and are chosen so that, if f' is taken as a new input, the system appears as a single unit mass.
- $m\ddot{x} + b\dot{x} + kx = \alpha f' + \beta.$

$$\alpha = m,$$
$$\beta = b\dot{x} + kx.$$





Control-Law Partitioning

$$\ddot{x} = f'.$$

$$f' = -k_v\dot{x} - k_p x.$$

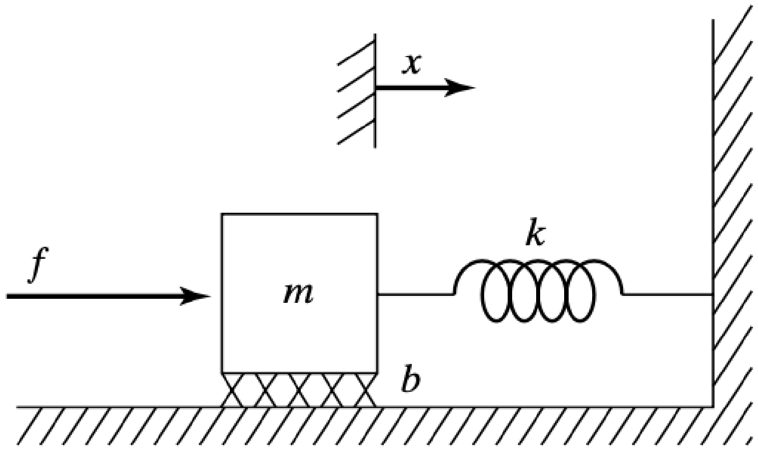
$$\ddot{x} + k_v\dot{x} + k_p x = 0.$$

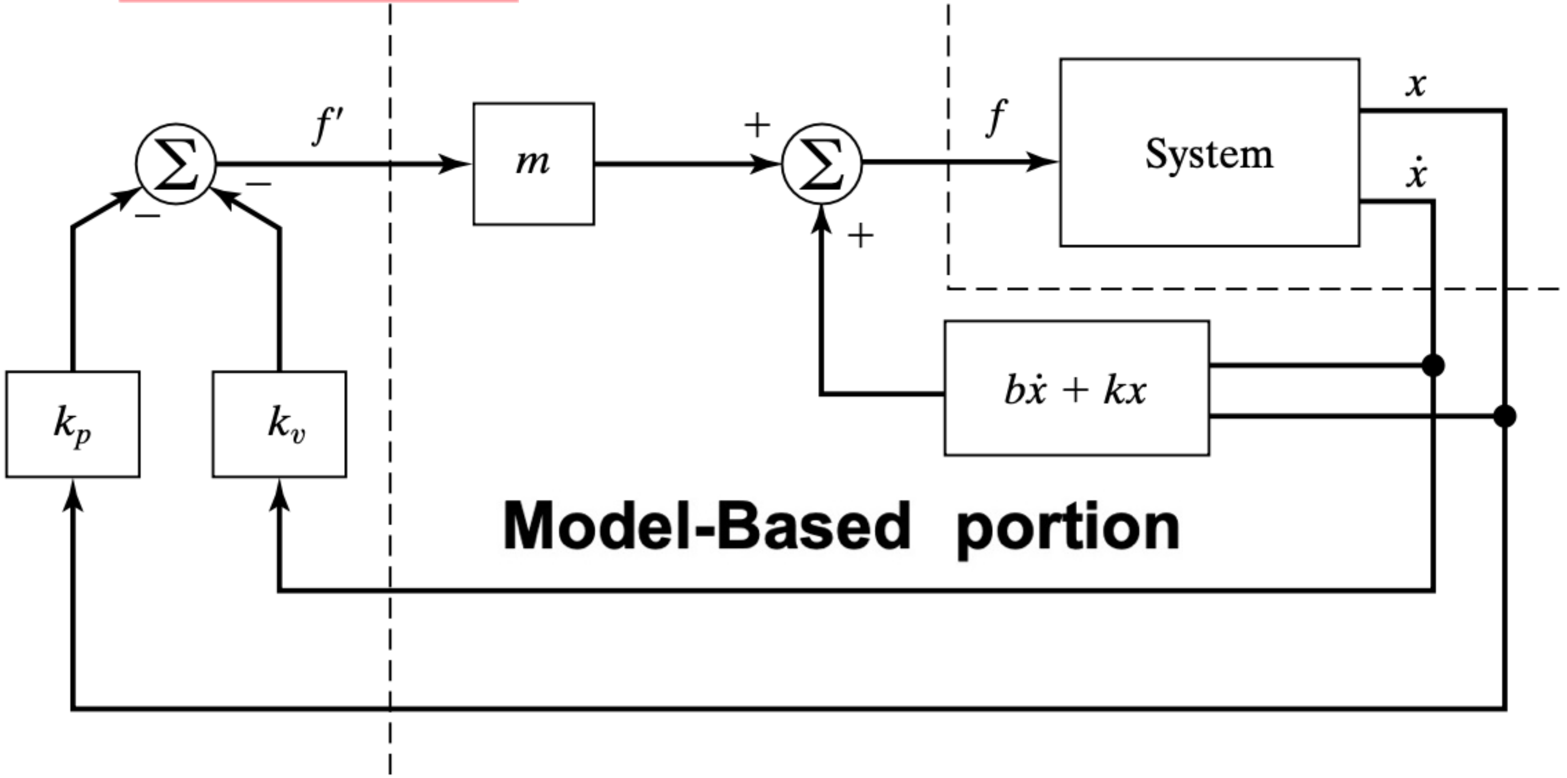
$$k_v = 2\sqrt{k_p}$$

$$\alpha = m,$$

$$\beta = b\dot{x} + kx.$$

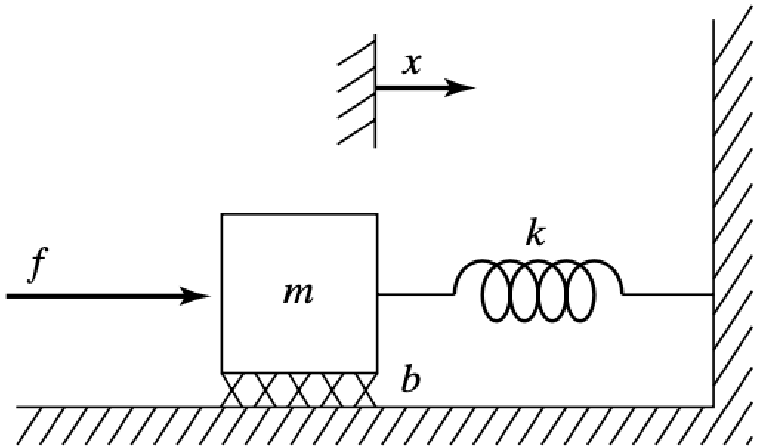
$$m\ddot{x} + b\dot{x} + kx = \alpha f' + \beta.$$





Control-Law Partitioning - Example

If the parameters of the system in Figure are $m = 1$, $b = 1$, and $k = 1$, find α, β, and the gains k_p and k_v for a position-regulation control law that results in the system's being critically damped with a closed-loop stiffness of 16.0.



Control-Law Partitioning - Example - Solution

If the parameters of the system in Figure are $m = 1$, $b = 1$, and $k = 1$, find α, β, and the gains k_p and k_v for a position-regulation control law that results in the system's being critically damped with a closed-loop stiffness of 16.0.

We choose

$$\alpha = 1,$$

$$\beta = \dot{x} + x,$$

so that the system appears as a unit mass from the fictitious f' input. We then set gain k_p to the desired closed-loop stiffness and set $k_v = 2\sqrt{k_p}$ for critical damping.

This gives

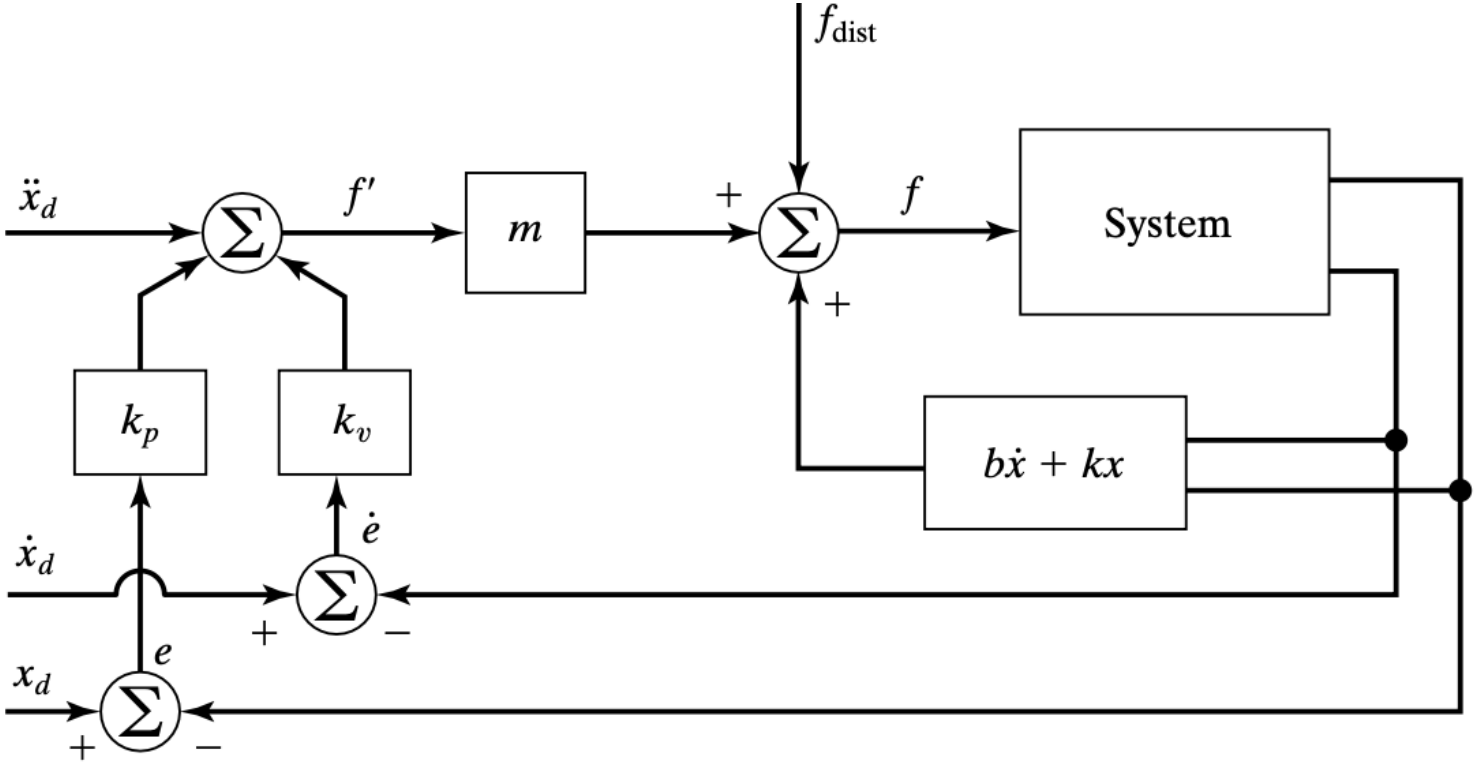
$$k_p = 16.0,$$

$$k_v = 8.0.$$

Disturbance Rejection

- $\ddot{e} + k_v\dot{e} + k_p e = f_{\text{dist}}$
- f_{dist}: bounded => the solution of $e(t)$ is also bounded
- Bounded input, bounded output (BIBO) stability

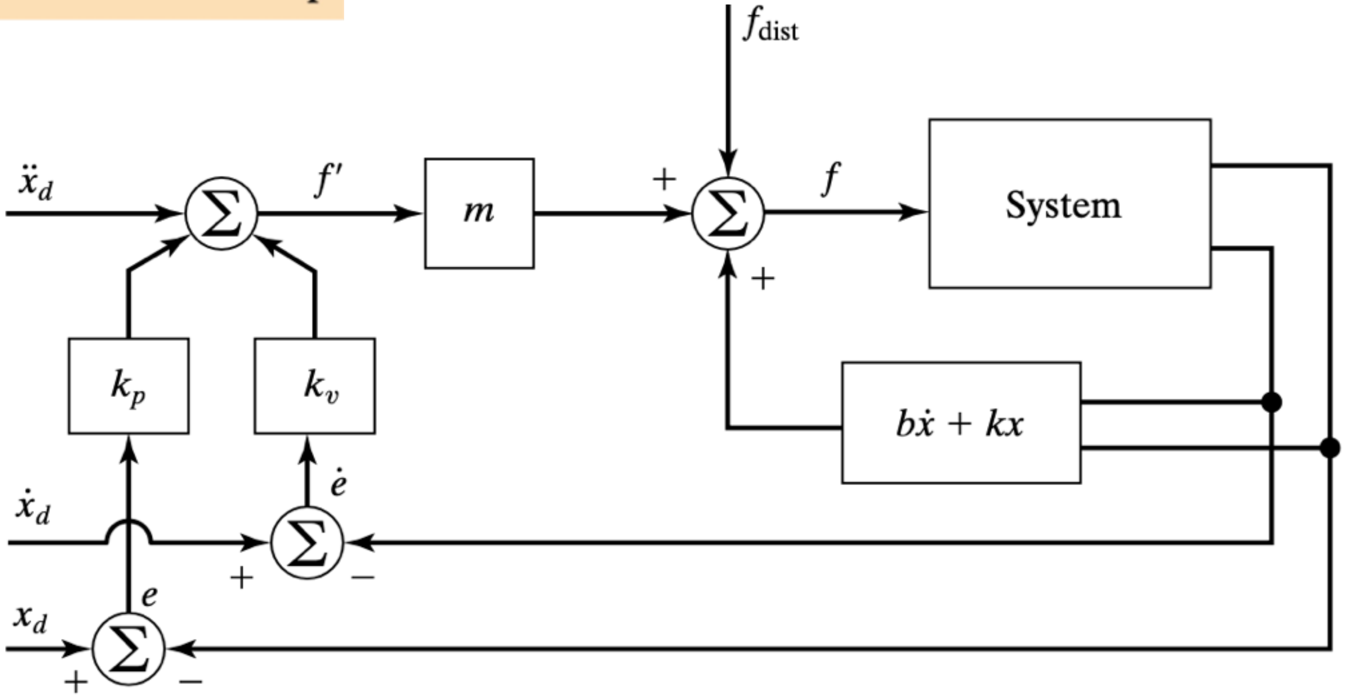
$$\max_t f_{\text{dist}}(t) < a,$$



Steady-state Error

- f_{dist}: constant
- **Steady-state analysis:** analyzing the system at rest (i.e. the derivatives of all system variables are zero)
- $e = f_{\text{dist}}/k_p$: **steady-state error**

$$\ddot{e} + k_v\dot{e} + k_p e = f_{\text{dist}}$$



Addition of an Integral term

- Addition of an integral term to eliminate steady-state error
- Modified control law: $f' = \ddot{x}_d + k_v \ddot{e} + k_p e + k_i \int e dt$
- Error equation: $\ddot{e} + k_v \dot{e} + k_p e + k_i \int e dt = f_{\text{dist}}$
- If $e(t)=0$ for $t<0$, for $t>0$ we have: $\ddot{e} + k_v \ddot{e} + k_p \dot{e} + k_i e = \dot{f}_{\text{dist}},$

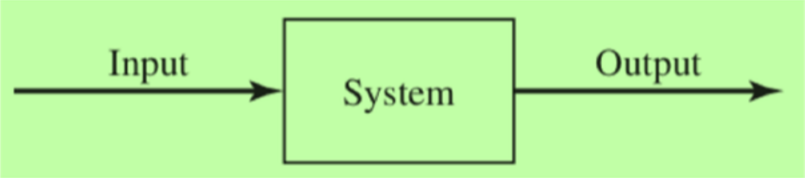
$k_i e = 0 \Rightarrow e = 0$: **steady-state error**

PID Control law (Proportional, Integral, Derivative)

Steps for controlling a given system

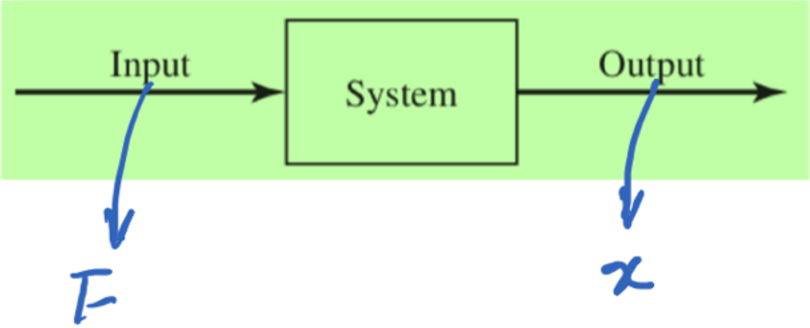
1. Dynamic model
2. Transfer Function
3. Evaluation / Analysis
4. Add Controller (P, PD, PI, PID) and close the loop

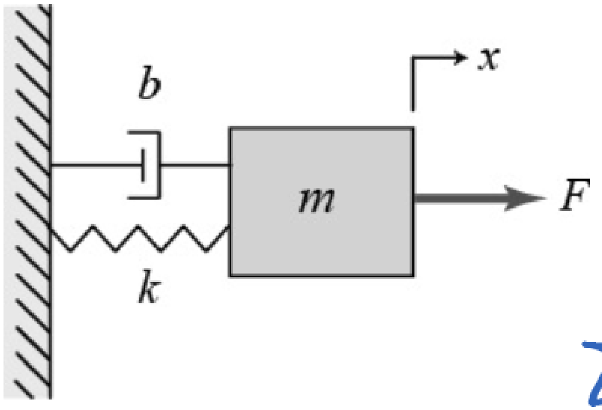
1. Dynamic Model

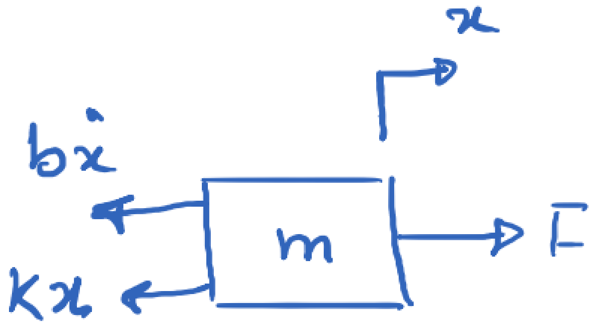


- We have a physical system with a specific behavior.
  - If you apply an input to the system, you will get an output
- We need Mathematical Equations describing the behavior of the system
  - What is the output given the input?
- These equations are called "Dynamic Model" of the system.
- Dynamic Model is in the form of "Differential Equations" and is in the "time" domain.

1. Dynamic Model - Example 1



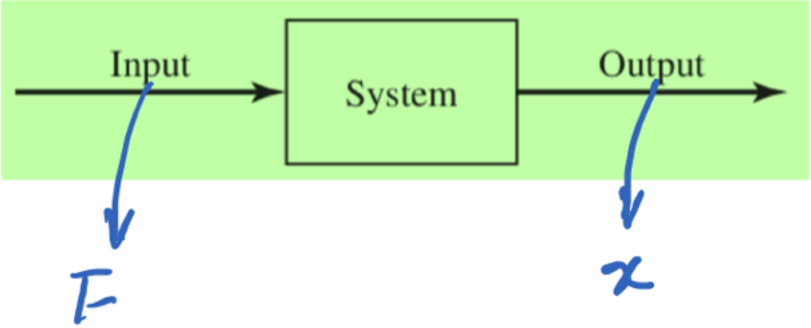


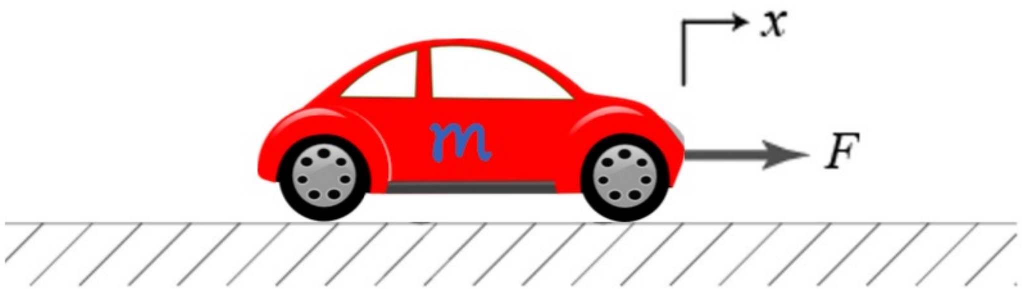


$$\Sigma F = ma \Rightarrow F - b\dot{x} - kx = m\ddot{x}$$

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

1. Dynamic Model - Example 2





$$\Sigma F = ma$$

$$\Rightarrow \boxed{F = m\ddot{x}}$$

2. Transfer Function (TF)

- From the dynamic model, there is a relation between output and input, and we need to know that.
- It is not straight-forward to derive this relation in the "time domain"
  - because of the derivative terms
- We need "Frequency Domain" because in the frequency domain, there is no derivatives.
- Laplace Transformation is used to represent the dynamical model in the "Frequency Domain".
- Derive the output/input relation (ratio of output/input) → This ratio is called "**Transfer Function**".

Time domain vs. Complex Frequency domain (s-domain)

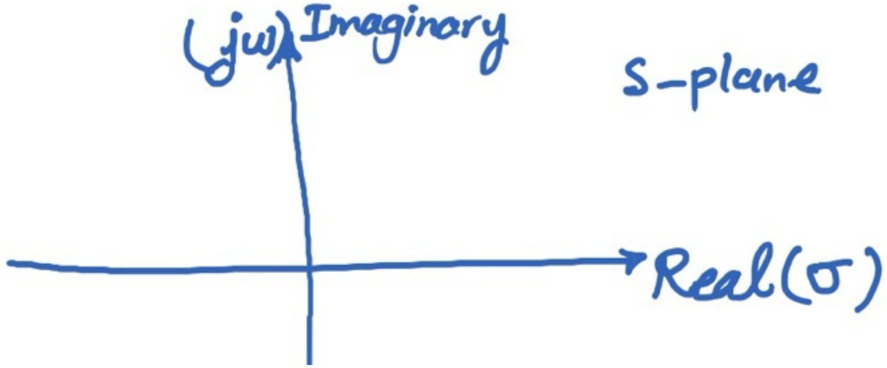
- System models are usually in the form of differential equations.
- Differential equations are in the "time domain" and include time derivatives. They are difficult to solve ☹
- Differential equations are composed of multiple terms/functions in time domain.
- For each function ***f(t)*** in time domain, there is a corresponding function in the complex frequency domain (***F(s)***)
- *f(t)* and *F(s)* are transformable and they both represent the same think but in different domains.
- *s* is a complex number located in the s-plane and depends on the system's frequency and damping.

s

$$F(s) = \int_0^\infty f(t)e^{-st}\,dt$$

$i = j = \sqrt{-1}$

$$s = \sigma + i\omega$$



https://www.youtube.com/watch?v=8QFwB0M7Ig8

Laplace Transformations

Table of Laplace Transforms

	$f(t)=\mathfrak{L}^{-1}\{F(s)\}$	$F(s)=\mathfrak{L}\{f(t)\}$		$f(t)=\mathfrak{L}^{-1}\{F(s)\}$	$F(s)=\mathfrak{L}\{f(t)\}$
1.	1	$\frac{1}{s}$	2.	$\mathbf{e}^{at}$	$\frac{1}{s-a}$
3.	$t^n,\quad n=1,2,3,\ldots$	$\frac{n!}{s^{n+1}}$	4.	$t^p, p > -1$	$\frac{\Gamma(p+1)}{s^{p+1}}$
5.	$\sqrt{t}$	$\frac{\sqrt{\pi}}{2s^{\frac{3}{2}}}$	6.	$t^{n-\frac{1}{2}},\quad n=1,2,3,\ldots$	$\frac{1\cdot 3\cdot 5\cdots(2n-1)\sqrt{\pi}}{2^n s^{n+\frac{1}{2}}}$
7.	$\sin(at)$	$\frac{a}{s^2+a^2}$	8.	$\cos(at)$	$\frac{s}{s^2+a^2}$
9.	$t\sin(at)$	$\frac{2as}{\left(s^2+a^2\right)^2}$	10.	$t\cos(at)$	$\frac{s^2-a^2}{\left(s^2+a^2\right)^2}$
11.	$\sin(at)-at\cos(at)$	$\frac{2a^3}{\left(s^2+a^2\right)^2}$	12.	$\sin(at)+at\cos(at)$	$\frac{2as^2}{\left(s^2+a^2\right)^2}$
13.	$\cos(at)-at\sin(at)$	$\frac{s\left(s^2-a^2\right)}{\left(s^2+a^2\right)^2}$	14.	$\cos(at)+at\sin(at)$	$\frac{s\left(s^2+3a^2\right)}{\left(s^2+a^2\right)^2}$
15.	$\sin(at+b)$	$\frac{s\sin(b)+a\cos(b)}{s^2+a^2}$	16.	$\cos(at+b)$	$\frac{s\cos(b)-a\sin(b)}{s^2+a^2}$
17.	$\sinh(at)$	$\frac{a}{s^2-a^2}$	18.	$\cosh(at)$	$\frac{s}{s^2-a^2}$

https://slideplayer.com/slide/14898687/

Laplace Transformations

Table of Laplace Transforms

	$f(t)=\mathfrak{L}^{-1}\{F(s)\}$	$F(s)=\mathfrak{L}\{f(t)\}$		$f(t)=\mathfrak{L}^{-1}\{F(s)\}$	$F(s)=\mathfrak{L}\{f(t)\}$
19.	$\mathbf{e}^{at}\sin(bt)$	$\dfrac{b}{(s-a)^2+b^2}$	20.	$\mathbf{e}^{at}\cos(bt)$	$\dfrac{s-a}{(s-a)^2+b^2}$
21.	$\mathbf{e}^{at}\sinh(bt)$	$\dfrac{b}{(s-a)^2-b^2}$	22.	$\mathbf{e}^{at}\cosh(bt)$	$\dfrac{s-a}{(s-a)^2-b^2}$
23.	$t^n\mathbf{e}^{at},\quad n=1,2,3,\ldots$	$\dfrac{n!}{(s-a)^{n+1}}$	24.	$f(ct)$	$\dfrac{1}{c}F\left(\dfrac{s}{c}\right)$
25.	$u_c(t)=u(t-c)$ Heaviside Function	$\dfrac{\mathbf{e}^{-cs}}{s}$	26.	$\delta(t-c)$ Dirac Delta Function	$\mathbf{e}^{-cs}$
27.	$u_c(t)f(t-c)$	$\mathbf{e}^{-cs}F(s)$	28.	$u_c(t)g(t)$	$\mathbf{e}^{-cs}\mathfrak{L}\{g(t+c)\}$
29.	$\mathbf{e}^{ct}f(t)$	$F(s-c)$	30.	$t^n f(t),\quad n=1,2,3,\ldots$	$(-1)^n F^{(n)}(s)$
31.	$\dfrac{1}{t}f(t)$	$\int_s^\infty F(u)\,du$	32.	$\int_0^t f(v)\,dv$	$\dfrac{F(s)}{s}$
33.	$\int_0^t f(t-\tau)g(\tau)\,d\tau$	$F(s)G(s)$	34.	$f(t+T)=f(t)$	$\dfrac{\int_0^T \mathbf{e}^{-st}f(t)\,dt}{1-\mathbf{e}^{-sT}}$
35.	$f'(t)$	$sF(s)-f(0)$	36.	$f''(t)$	$s^2F(s)-sf(0)-f'(0)$
37.	$f^{(n)}(t)$	$s^nF(s)-s^{n-1}f(0)-s^{n-2}f'(0)\cdots-sf^{(n-2)}(0)-f^{(n-1)}(0)$			

https://slideplayer.com/slide/14898687/

Laplace Transformations - Properties

$f(t)=L^{-1}\{F(s)\}$	$F(s)=L\{f(t)\}$
$e^{at}f(t)$	$F(s-a)$
$\int_0^t f(\tau)d\tau$	$\frac{F(s)}{s}$
$f'(t) = \dot{f}(t)$	$\boxed{sF(s)-\cancel{f(0)}}$
$f''(t) = \ddot{f}(t)$	$\boxed{s^2F(s)-s\cancel{f(0)}-\cancel{f'(0)}}$
$f'''(t)$	$s^3F(s)-s^2f(0)-sf'(0)-f''(0)$

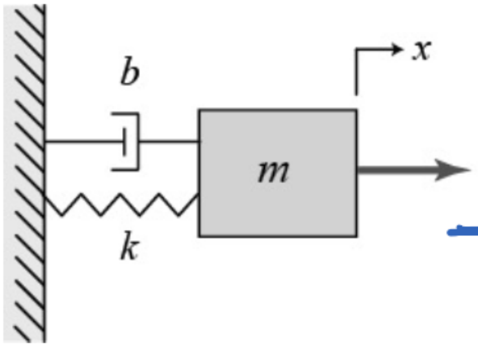
Initial conditions:

$f(0)=0$, $\dot{f}(0)=0$

$f(t)=x(t)$

https://slideplayer.com/slide/14898687/

2. Transfer Function - Example 1



$$\underbrace{m\ddot{x}}+\underbrace{b\dot{x}}+\underbrace{kx}=\underline{\underline{F(t)}}$$

$$\xrightarrow{\mathcal{L}} ms^2X(s)+bsX(s)+kX(s)=F(s)$$

$$\Rightarrow X(s)\left[ms^2+bs+k\right]=F(s)$$

$$\Rightarrow \boxed{TF=\frac{X(s)}{F(s)}=\frac{1}{ms^2+bs+k}}$$

output $\rightarrow x$

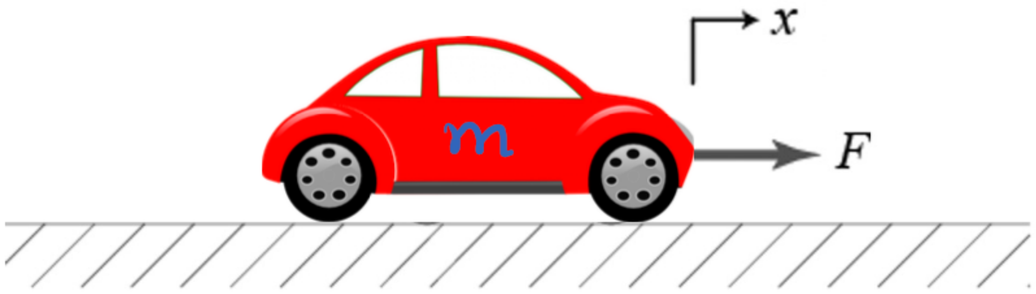
input $\rightarrow F$ $\Rightarrow TF=\dfrac{X(s)}{F(s)}$

$$\dot{x}(t)\xrightarrow{\mathcal{L}} sX(s)$$

$$\ddot{x}(t)\xrightarrow{\mathcal{L}} s^2X(s)$$

63

2. Transfer Function - Example 2



$$m\ddot{x} = F$$

$$\xrightarrow{\mathcal{L}} m s^2 X(s) = F(s)$$

$$\Rightarrow \boxed{TF = \frac{X(s)}{F(s)} = \frac{1}{ms^2}}$$

3. Evaluation/Analysis

- Having the Transform Function (TF), we can conclude about the **performance** (transient and steady-state response) and ability of the system.
- To evaluate the performance, we use tests inputs (input signals) to see how the system reacts to tests inputs.
- The response of the system to the test inputs replicates the response of the system to real world inputs.
- There are multiple tests inputs. Here are three of them:

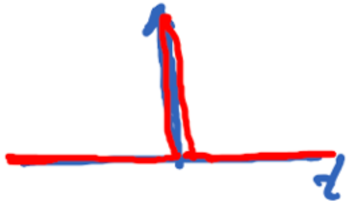
  - Unit Impulse: $u(t) = \delta(t) \rightarrow U(s) = 1$
  - Unit Step: $u(t) = 1 \rightarrow U(s) = \frac{1}{s}$
  - Unit Ramp: $u(t) = t \rightarrow U(s) = \frac{1}{s^2}$

3. Evaluation/Analysis

- Unit Impulse: $u(t) = \delta(t) \rightarrow U(s) = 1$

$$\delta(t) = \begin{cases} 0 & t \neq 0 \\ \infty & t = 0 \end{cases}$$

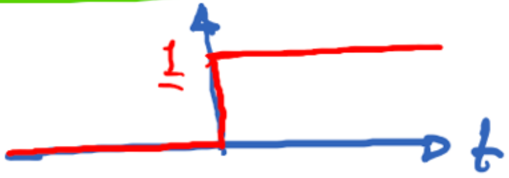
$$\boxed{U(s) = 1}$$



- Unit Step: $u(t) = 1 \rightarrow U(s) = \frac{1}{s}$

$u(t) = 1 \quad t \geq 0$

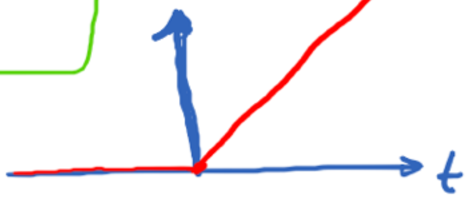
$$\boxed{U(s) = \frac{1}{s}}$$



- Unit Ramp: $u(t) = t \rightarrow U(s) = \frac{1}{s^2}$

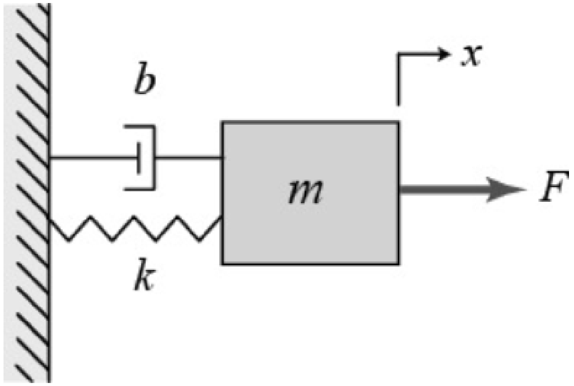
$u(t) = t \quad t \geq 0$

$$\boxed{U(s) = \frac{1}{s^2}}$$



3. Evaluation/Analysis - Example 1

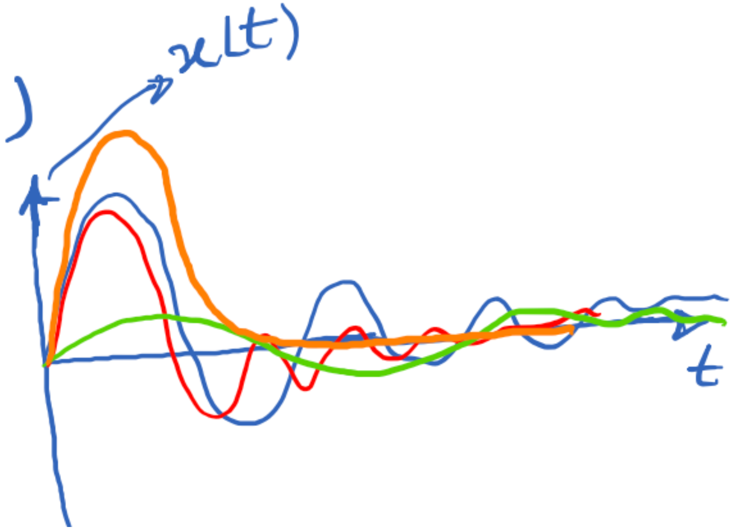
- Unit Impulse: $u(t) = \delta(t) \rightarrow U(s) = 1$



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

$$\boxed{U(s) = 1}$$

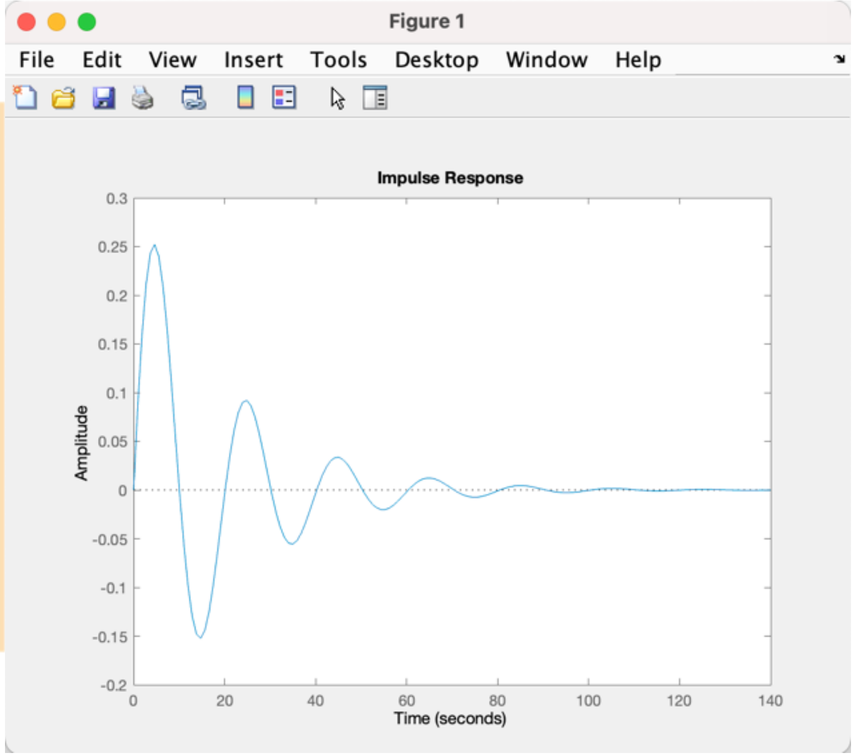
$F(s) = U(s)$

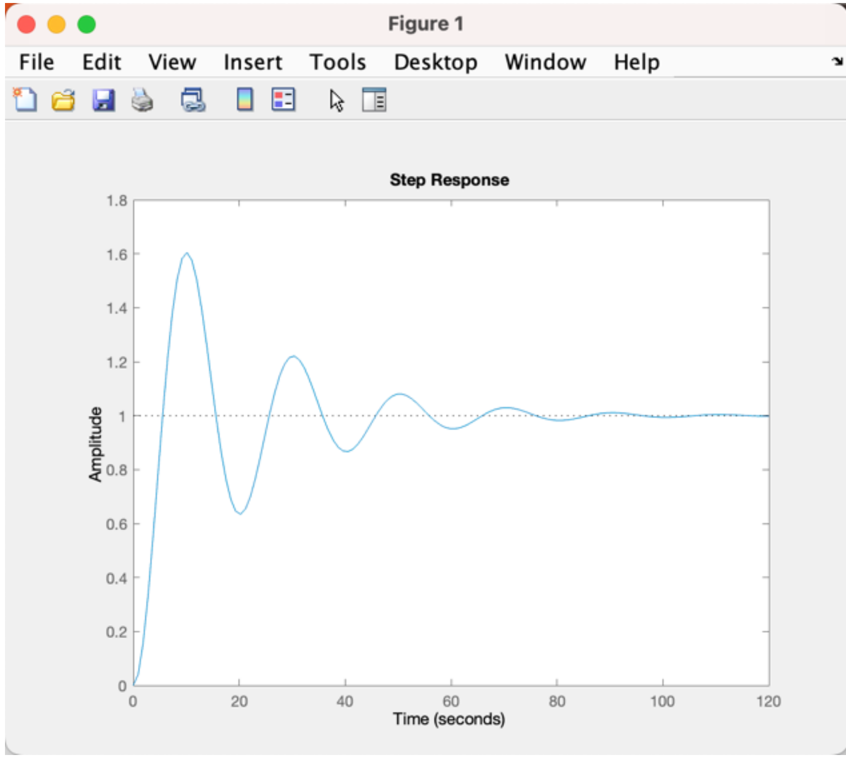


3. Evaluation/Analysis - Example 1 - in MATLAB

- Unit Impulse: $u(t) = \delta(t) \rightarrow U(s) = 1$

```
>> sys = tf([1],[10 1 1])

sys =

        1
  --------------
  10 s^2 + s + 1

Continuous-time transfer function.

>> impulse(sys)
>> step(sys)
```





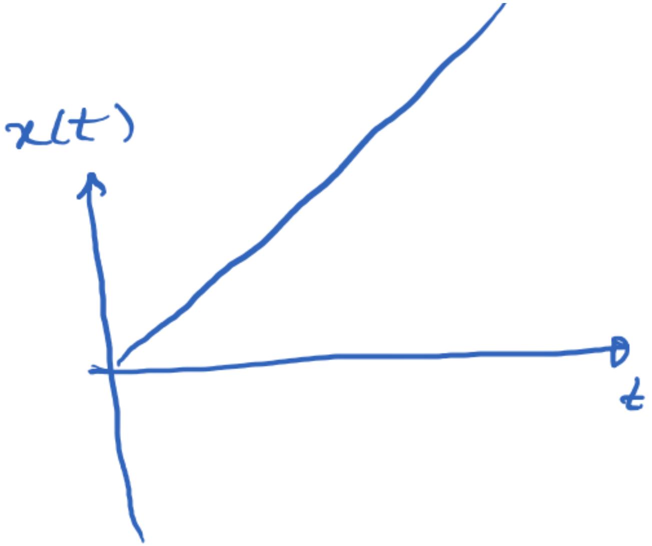
- Unit Step: $u(t) = 1 \rightarrow U(s) = \frac{1}{s}$

3. Evaluation/Analysis - Example 2

- Unit Impulse: $u(t) = \delta(t) \rightarrow U(s) = 1$

$$TF = \frac{X(s)}{U(s)} = \frac{1}{ms^2}$$

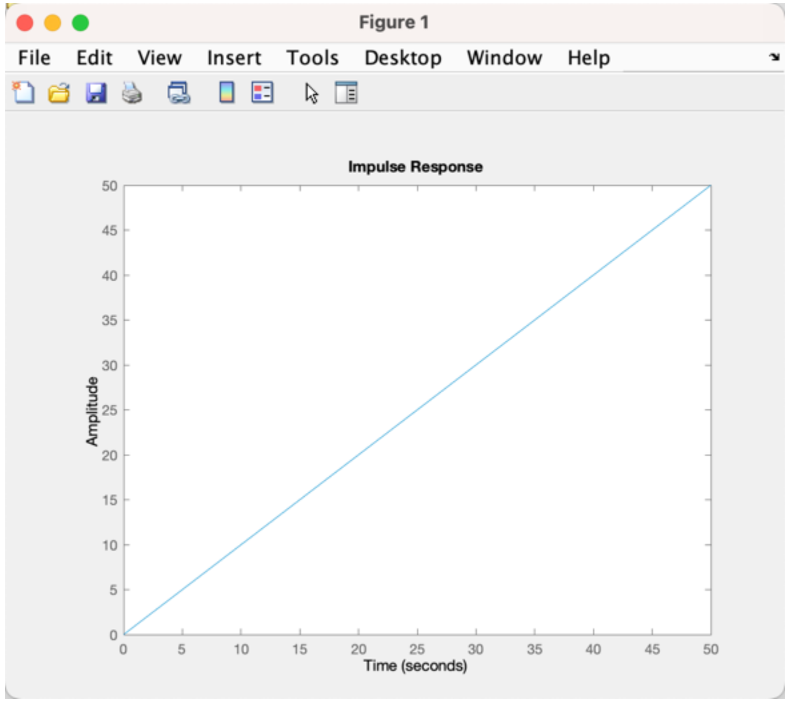
$$\underline{U(s) = 1}$$

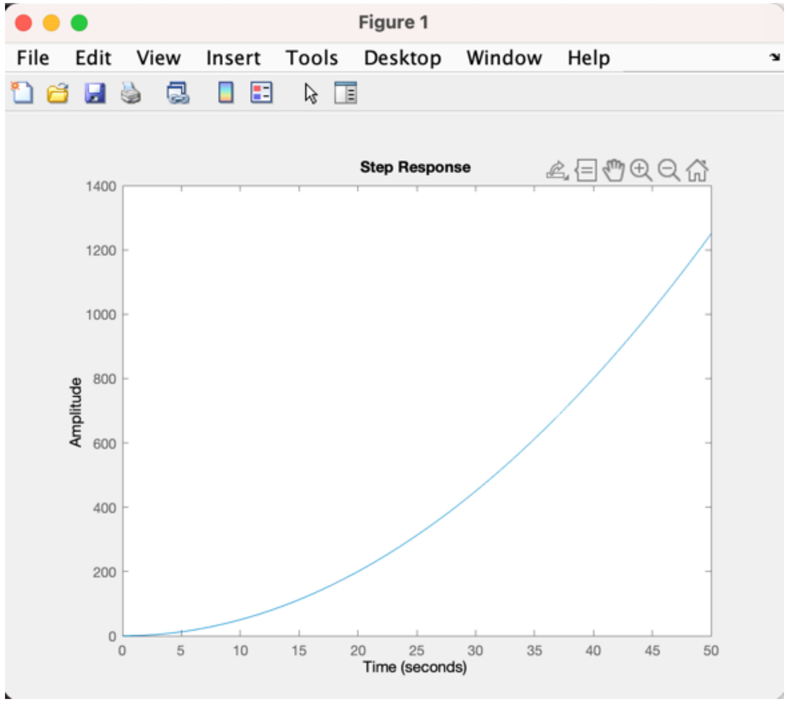


3. Evaluation/Analysis - Example 2 - in MATLAB

- Unit Impulse: $u(t) = \delta(t) \rightarrow U(s) = 1$

```
>> sys = tf([1], [1 0 0])

sys =

   1
  ---
  s^2

Continuous-time transfer function.

>> impulse(sys)
>> step(sys)
```





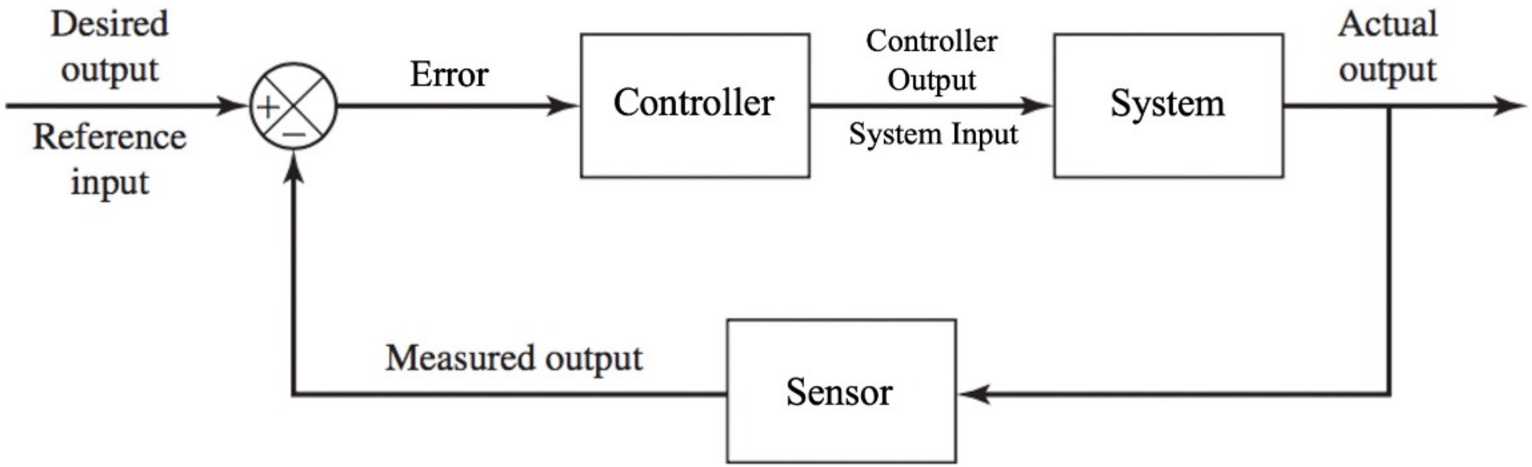
- Unit Step: $u(t) = 1 \rightarrow U(s) = \frac{1}{s}$

Classical Control - NOT satisfactory ☹

- Almost always, the performance of the system is NOT satisfactory and that is why we add a controller to the system.

4. Add Controller (P, PD, PI, PID) and close the loop

- Depending on the control goal, add a controller and find the Closed-Loop Transfer Function (CLTF) of the system.
- By adding a controller, the output/response/behavior of the system will change (why?)
- Repeat Step 3 (Evaluate/Analysis) and Step 4 (Add a controller) until a satisfactory performance is achieved.



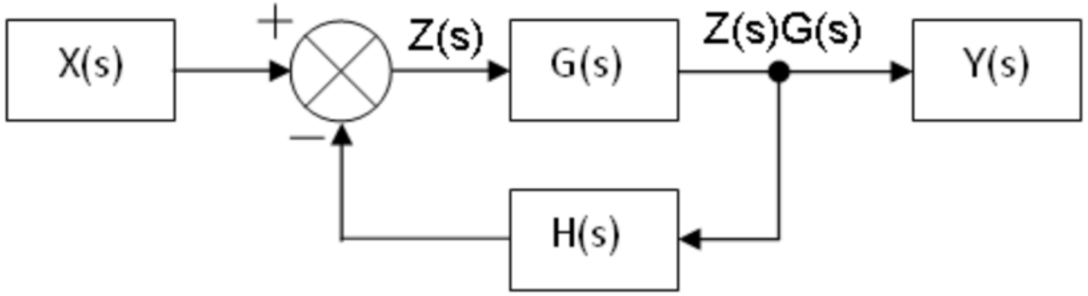
Closed-Loop Transfer Function (CLTF)

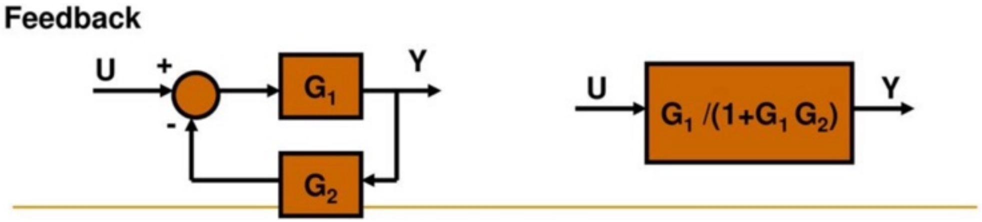
In control theory, a **closed-loop transfer function** is a mathematical function describing the net result of the effects of a feedback control loop on the input signal to the plant under control.

Overview [edit]

The closed-loop transfer function is measured at the output. The output signal can be calculated from the closed-loop transfer function and the input signal. Signals may be waveforms, images, or other types of data streams.

An example of a closed-loop transfer function is shown below:





The summing node and the *G*(*s*) and *H*(*s*) blocks can all be combined into one block, which would have the following transfer function:

$$\frac{Y(s)}{X(s)} = \frac{G(s)}{1 + G(s)H(s)}$$

$G(s)$ is called feedforward transfer function, $H(s)$ is called feedback transfer function, and their product $G(s)H(s)$ is called the **open-loop transfer function**.

https://en.wikipedia.org/wiki/Closed-loop_transfer_function

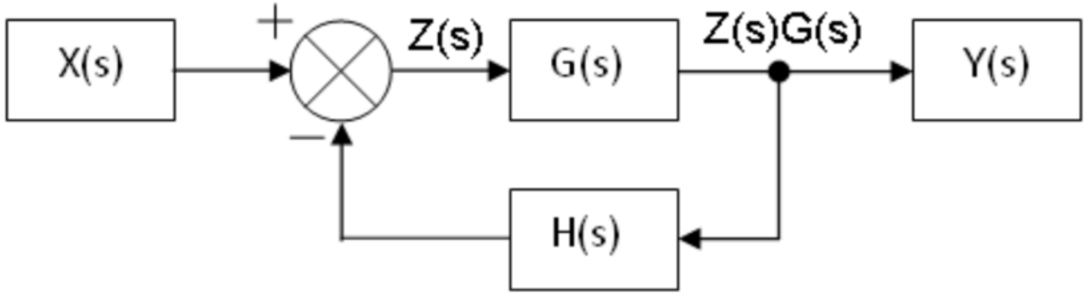
Closed-Loop Transfer Function (CLTF)

In control theory, a **closed-loop transfer function** is a mathematical function describing the net result of the effects of a feedback control loop on the input signal to the plant under control.

Overview [edit]

The closed-loop transfer function is measured at the output. The output signal c
Signals may be waveforms, images, or other types of data streams.

An example of a closed-loop transfer function is shown below:



The summing node and the $G(s)$ and $H(s)$ blocks can all be combined into one b

$$\frac{Y(s)}{X(s)} = \frac{G(s)}{1 + G(s)H(s)}$$

Derivation [edit]

We define an intermediate signal Z (also known as error signal) shown as follows:

Using this figure we write:

$$Y(s) = G(s)Z(s)$$

$$Z(s) = X(s) - H(s)Y(s)$$

Now, plug the second equation into the first to eliminate Z(s):

$$Y(s) = G(s)[X(s) - H(s)Y(s)]$$

Move all the terms with Y(s) to the left hand side, and keep the term with X(s) on the right hand side:

$$Y(s) + G(s)H(s)Y(s) = G(s)X(s)$$

Therefore,

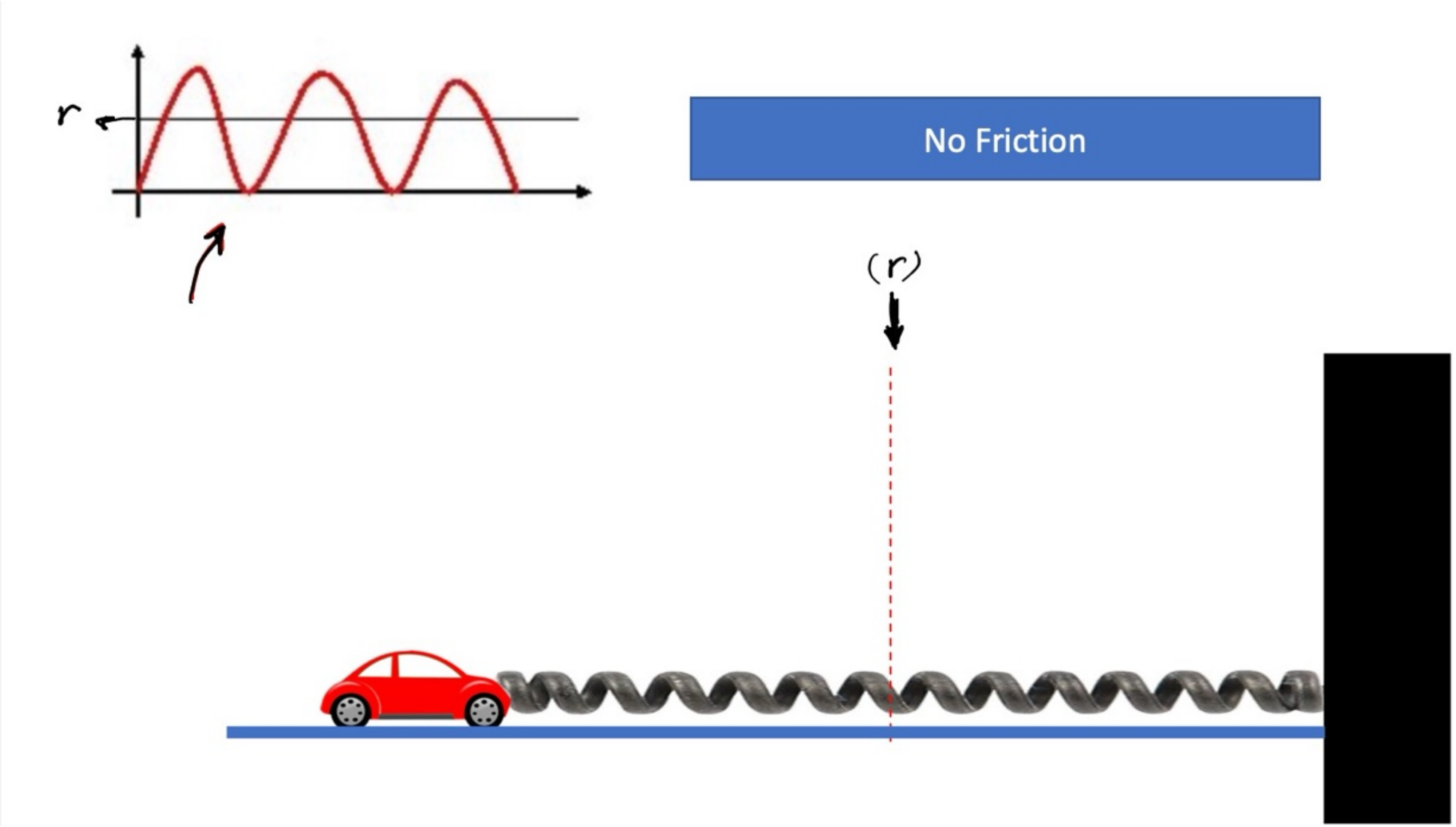
$$Y(s)(1 + G(s)H(s)) = G(s)X(s)$$

$$\Rightarrow \frac{Y(s)}{X(s)} = \frac{G(s)}{1 + G(s)H(s)}$$

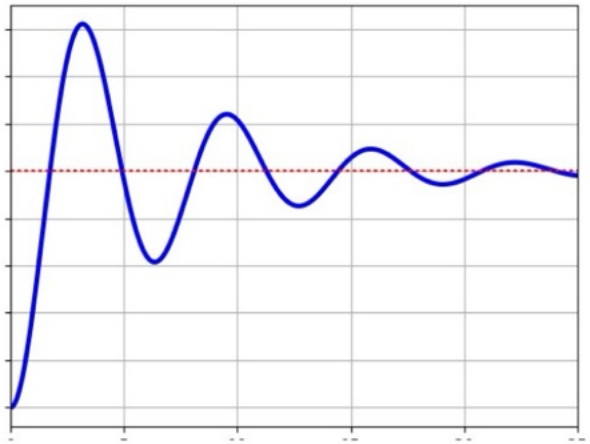
$G(s)$ is called feedforward transfer function, $H(s)$ is called feedback transfer function, and their product $G(s)H(s)$ is called the **open-loop transfer function**.

https://en.wikipedia.org/wiki/Closed-loop_transfer_function

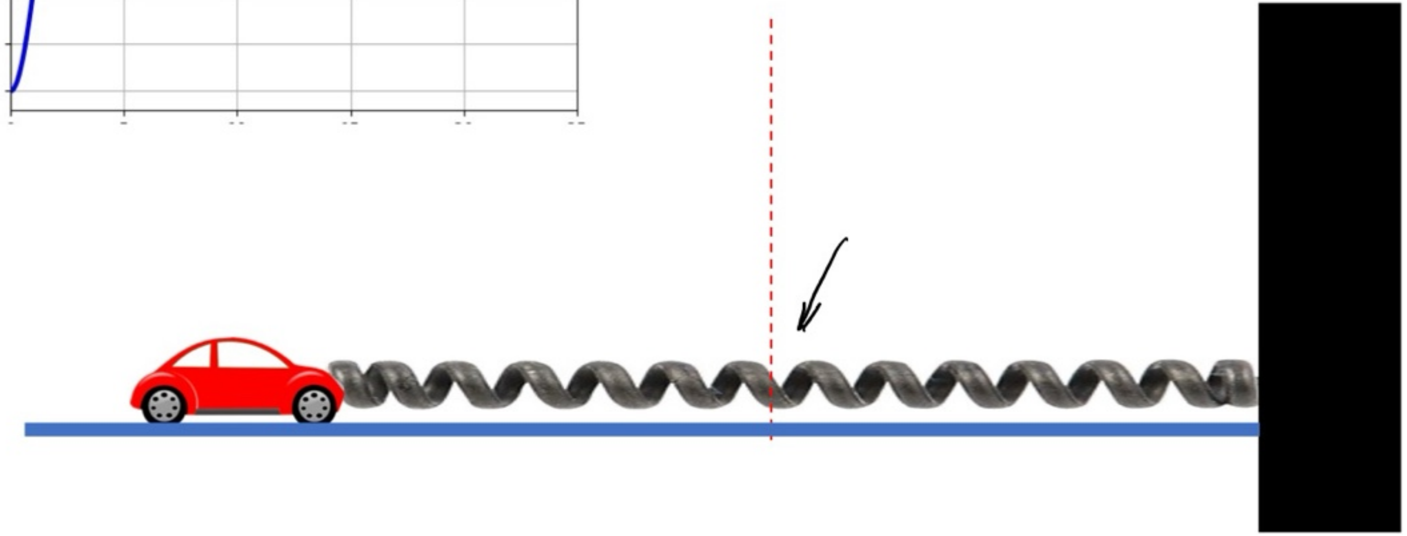
Proportional Controller



Proportional Controller



With Friction



Proportional Controller

- **Adding a Proportional Controller (K_p) is equivalent to adding a Spring!**

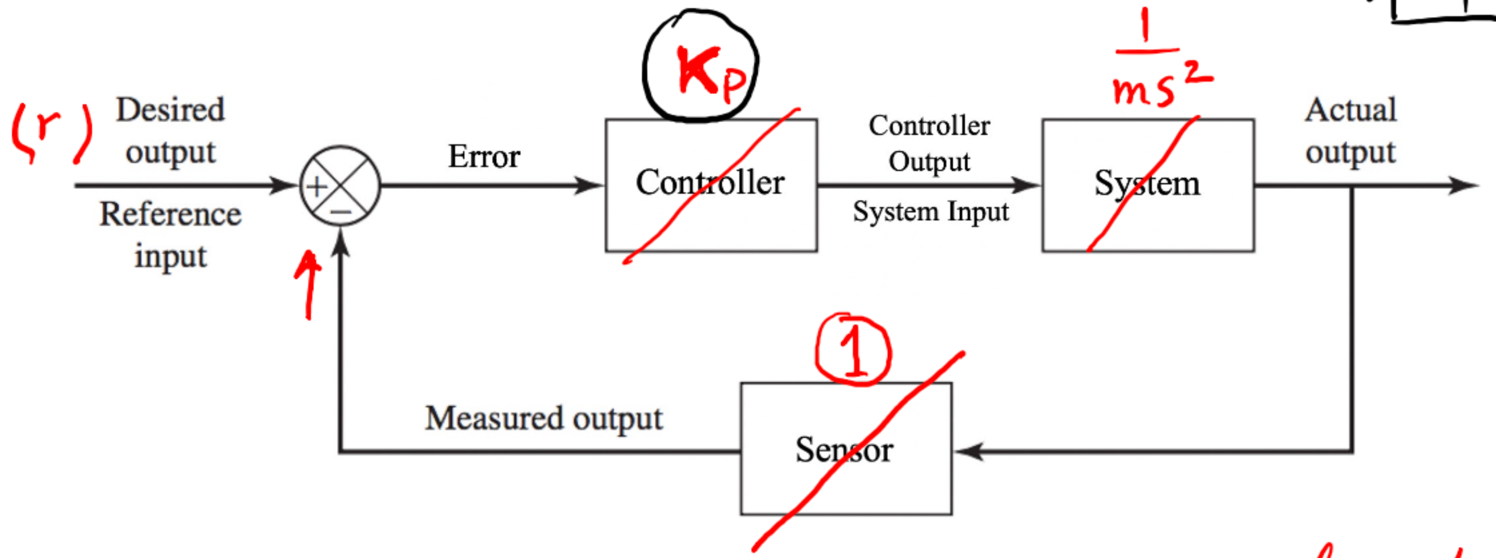
Proportional Controller

proportional controller: K_p

$\xrightarrow{e} \boxed{K_p} \xrightarrow{F} \Rightarrow \boxed{F = K_p e}$

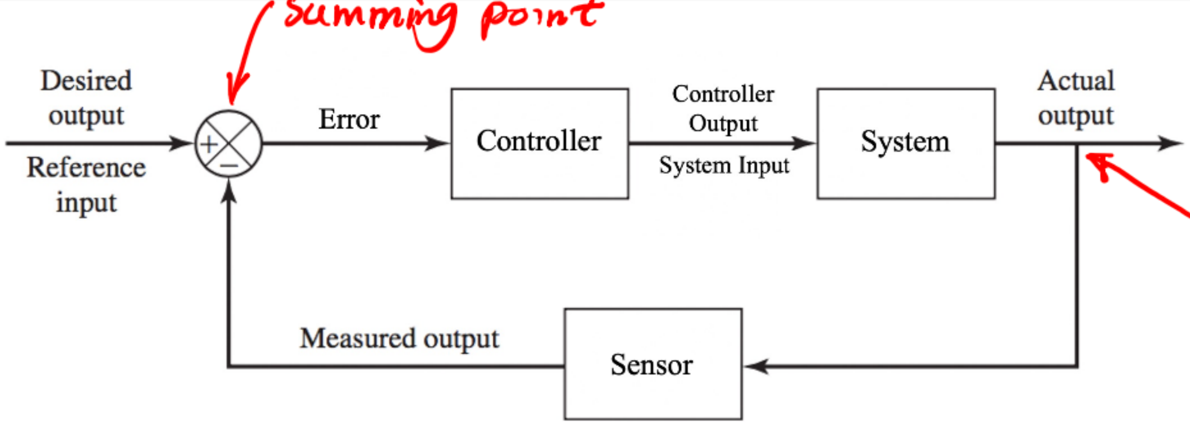
$E(s) \rightarrow \boxed{K_p} \xrightarrow{F(s)} \Rightarrow F(s) = K_p \cdot E(s)$

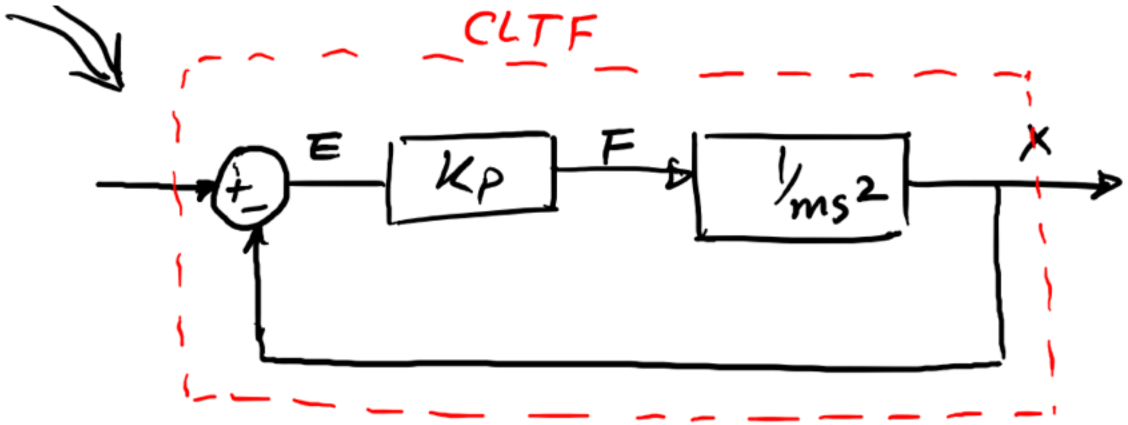
$\Rightarrow K_p = \dfrac{F(s)}{E(s)}$

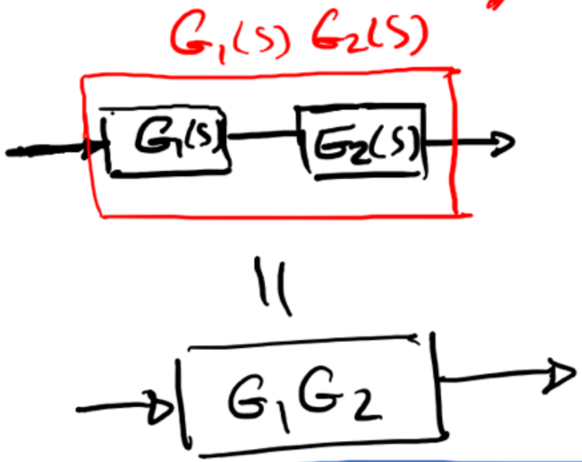


unity-feedback control systems

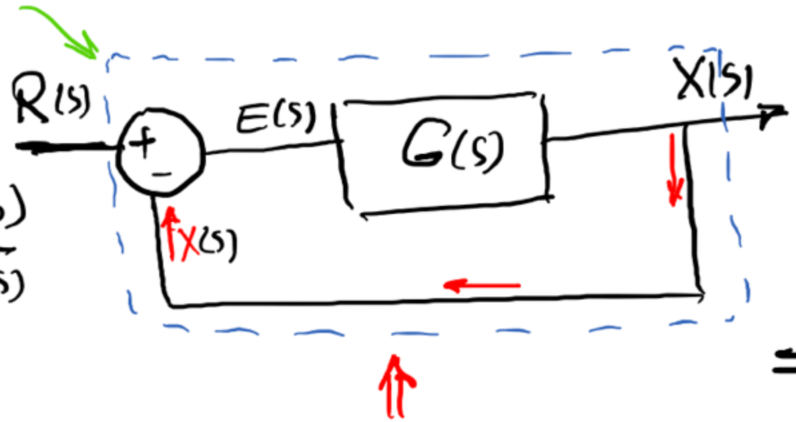
Proportional Controller







$$TF = \frac{X(s)}{R(s)}$$

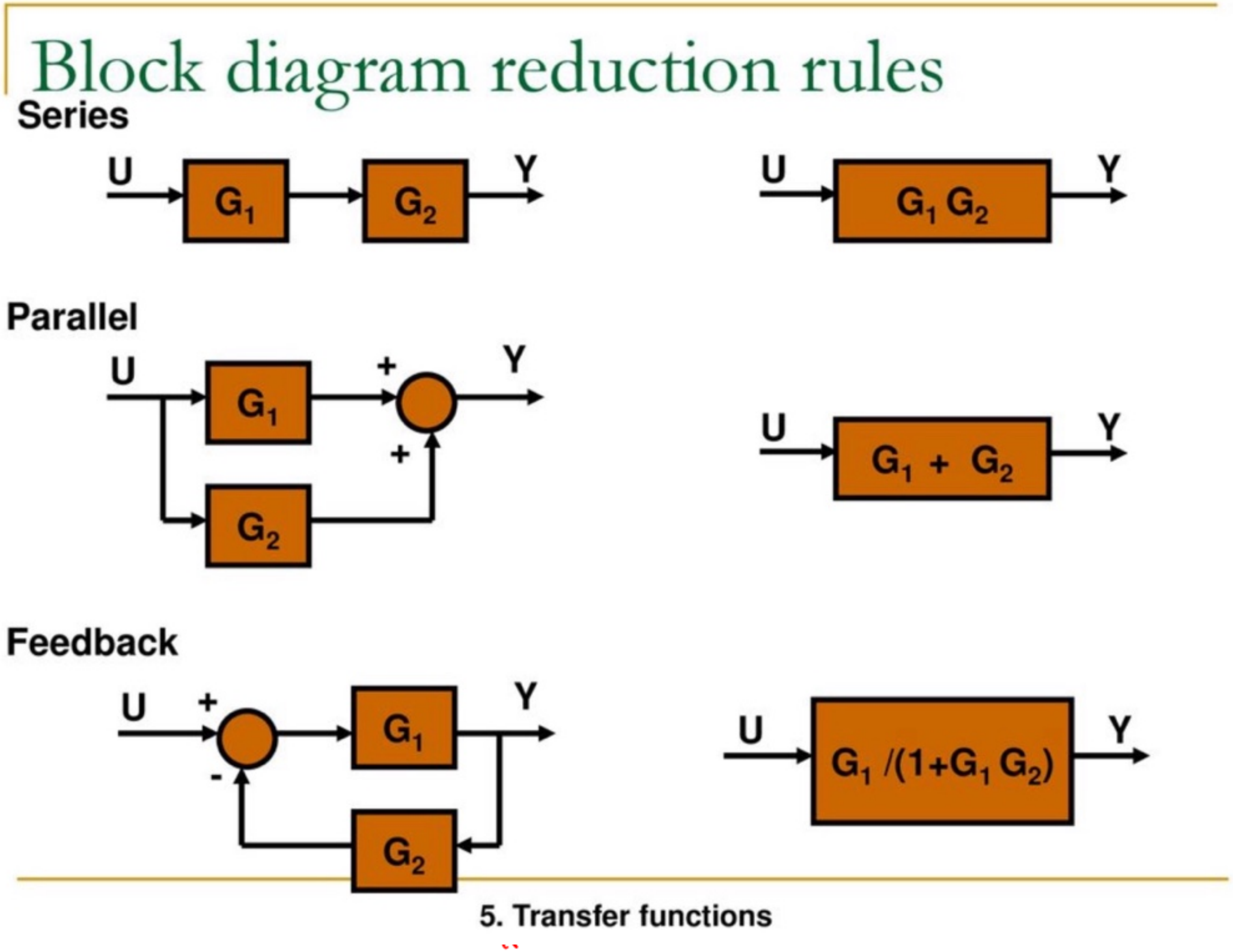


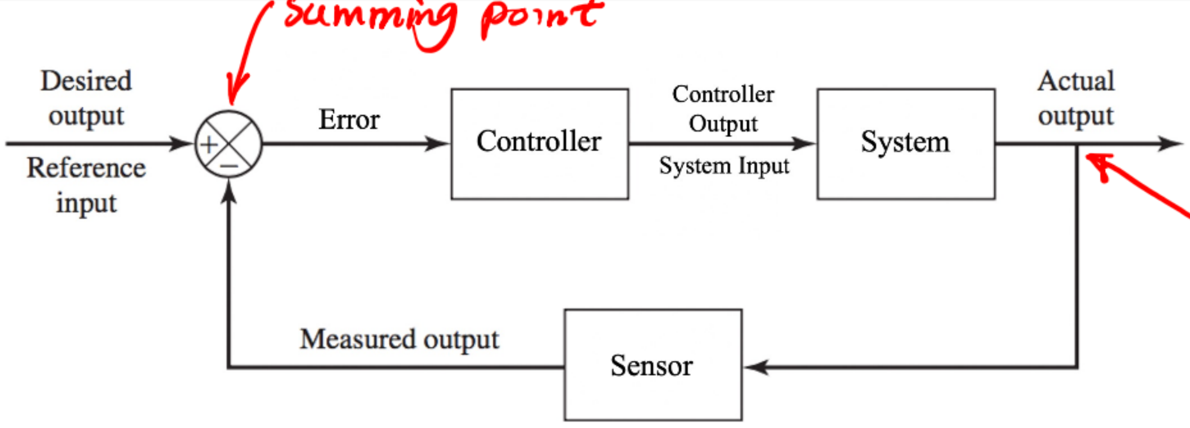
$$R(s) - X(s) = E(s)$$

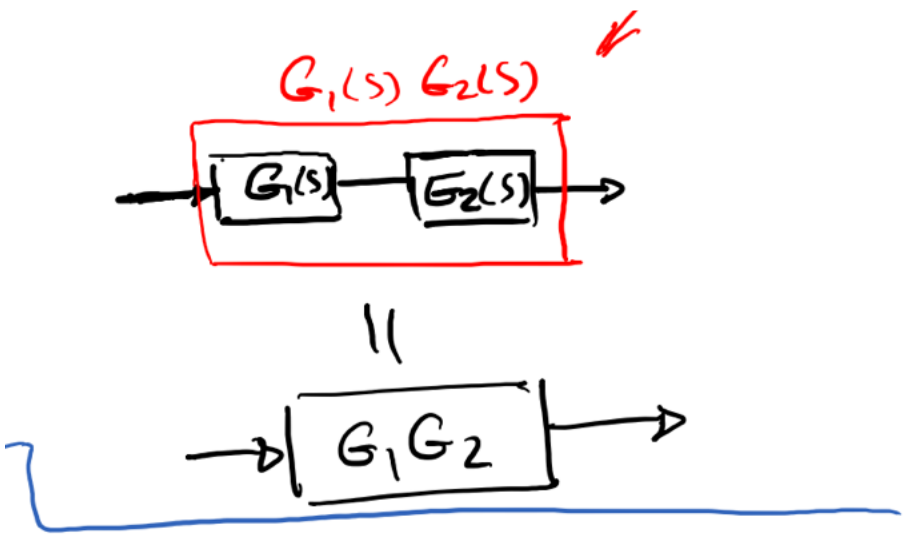
$$E(s)\,G(s) = X(s) \Rightarrow (R(s) - X(s))\,G(s) = X(s)$$

$$\Rightarrow R(s)G(s) - X(s)G(s) = X(s) \Rightarrow \boxed{\frac{X(s)}{R(s)} = \frac{G(s)}{1+G(s)}}$$

Proportional Controller





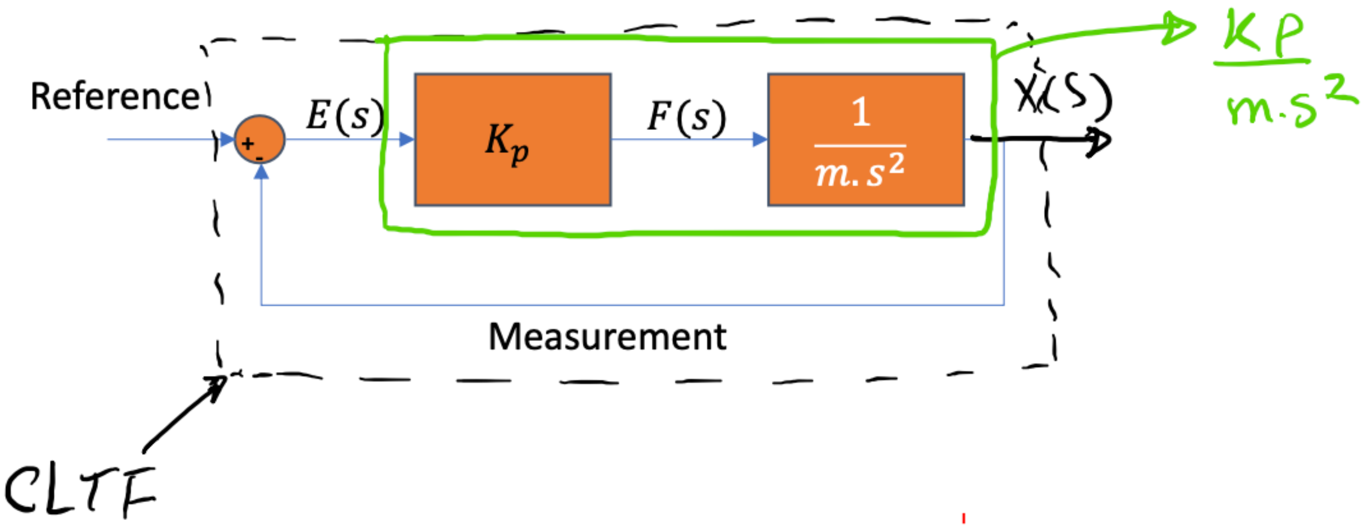


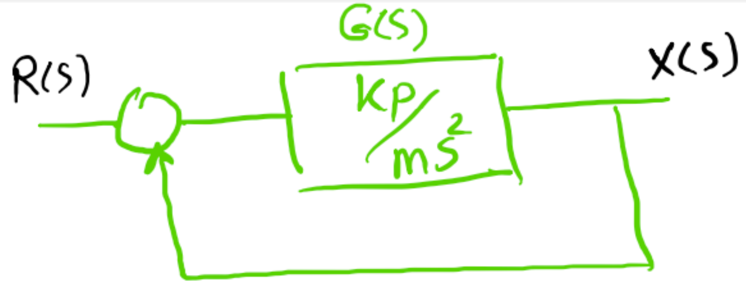
$E(S)$

$S) \Rightarrow (R(S) - X(S))\,G(S) = X(S)$

$(S)\,G(S) = X(S) \Rightarrow \boxed{\dfrac{X(S)}{R(S)} = \dfrac{G(S)}{1+G(S)}}$

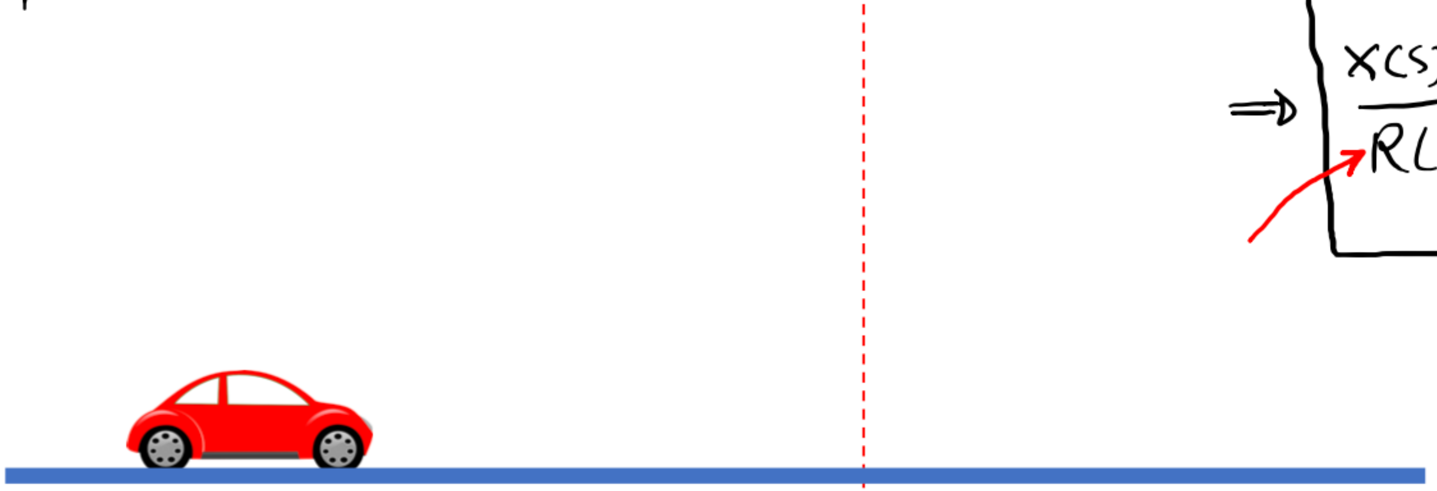
Proportional Controller





$$\frac{X(S)}{R(S)} = \frac{Kp/ms^2}{1 + Kp/ms^2}$$

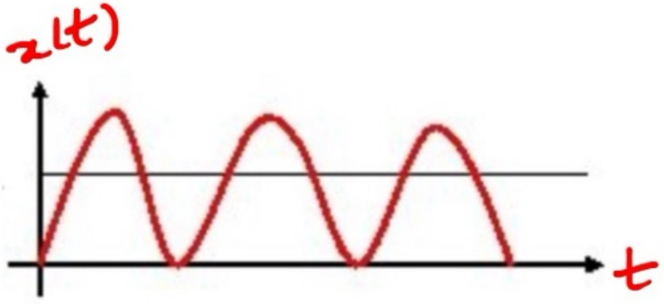
$$\Rightarrow \boxed{\frac{X(S)}{R(S)} = CLTF = \frac{Kp}{ms^2 + Kp}}$$

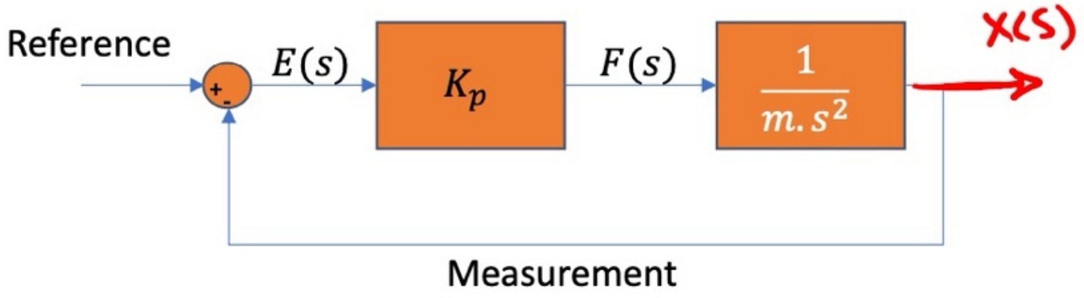


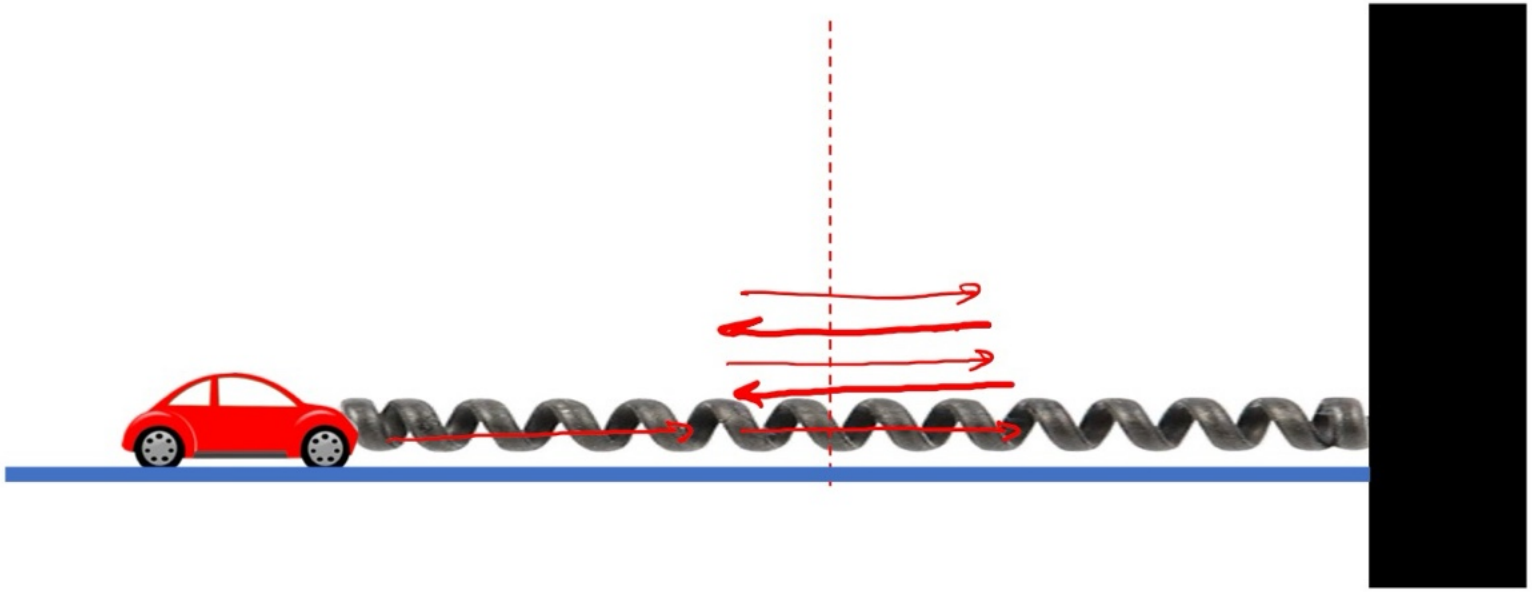
$m = 500$

$Kp > 0$

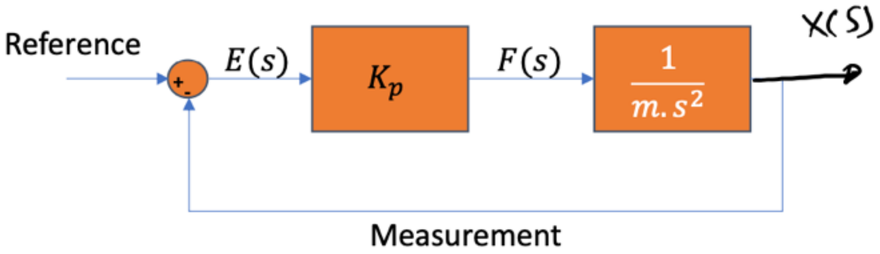
Proportional Controller



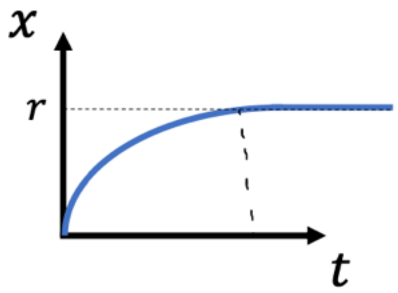




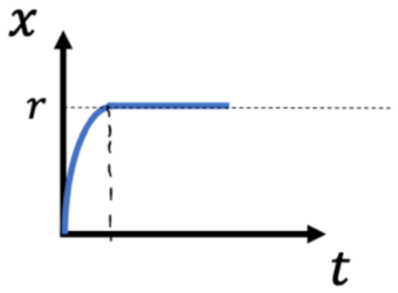
Derivative Controller



- By trying different values of Kp, we might get:

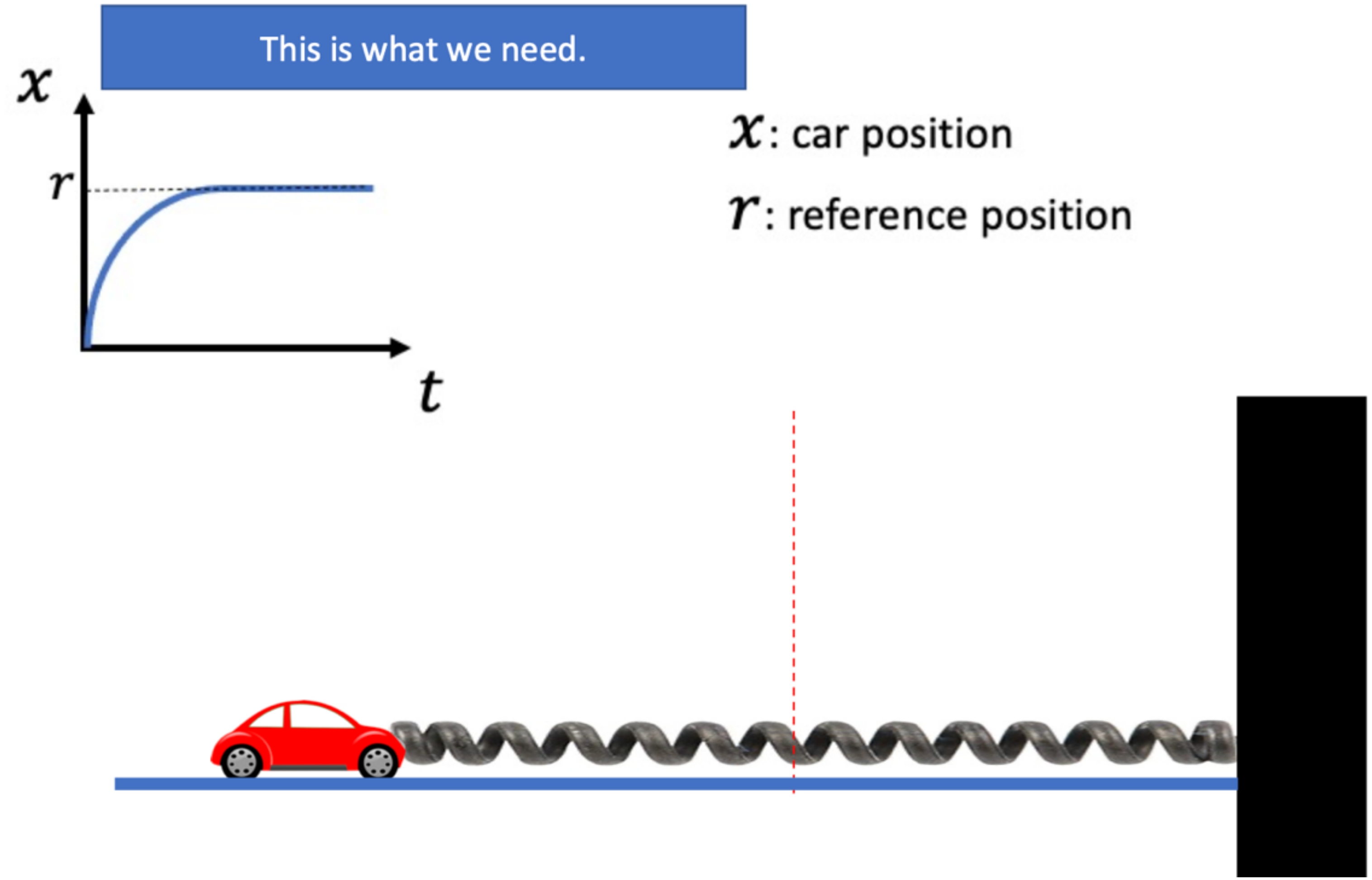


- But what if we want:



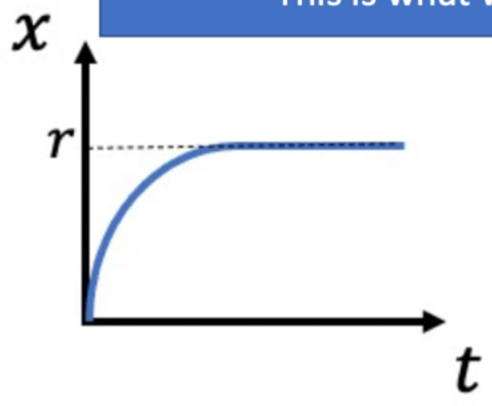
We need different strategies

Derivative Controller



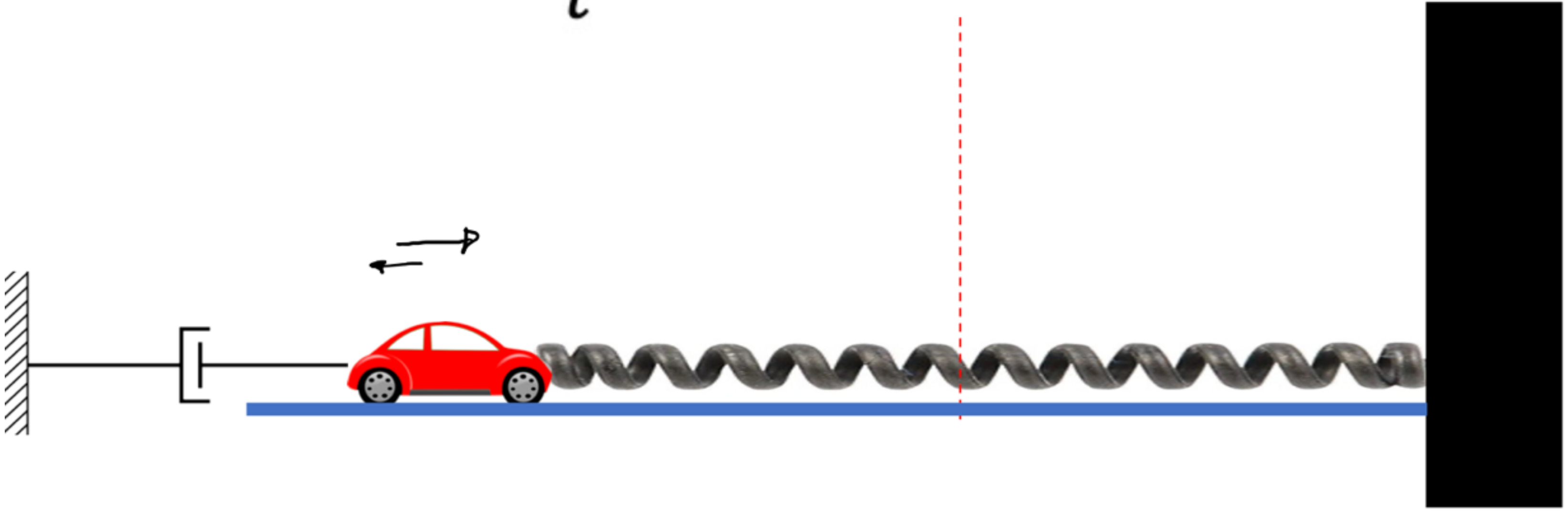
Derivative Controller

This is what we need.

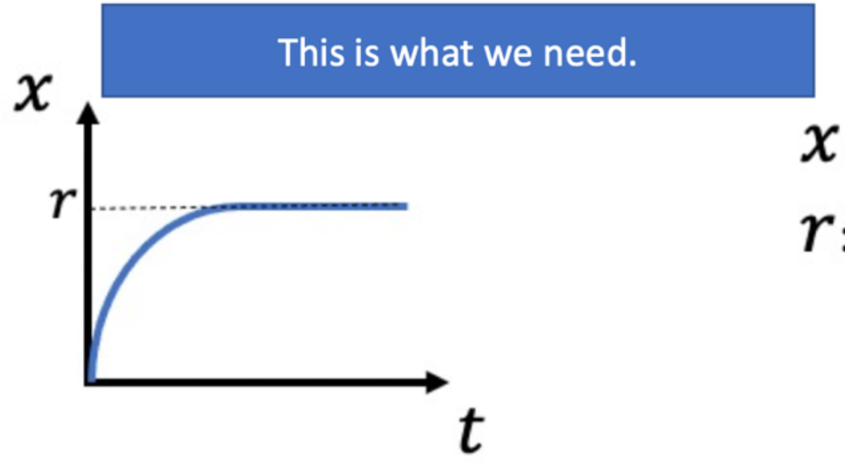


x: car position

r: reference position

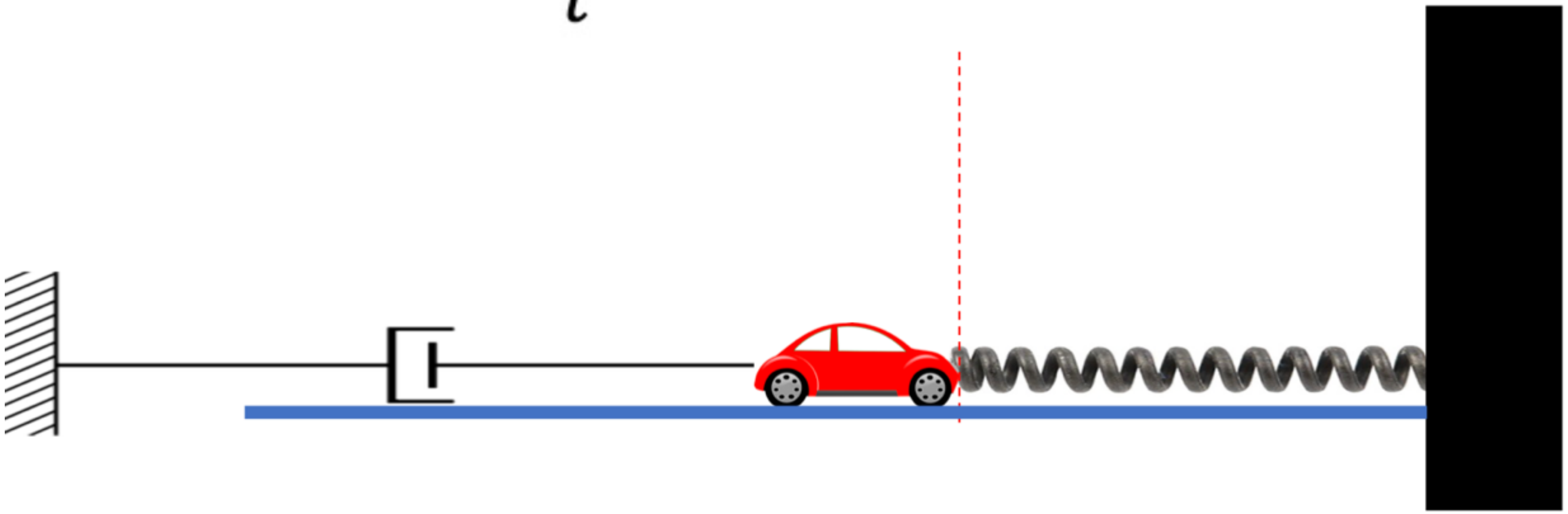


Derivative Controller



x: car position

r: reference position

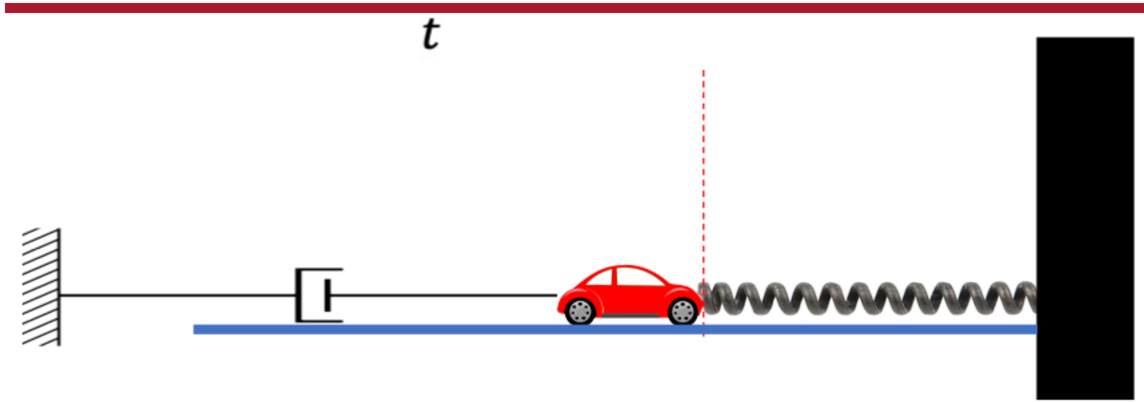


Derivative Controller

- **Adding a Derivative Controller ($K_d . s$) is equivalent to adding a Damper!**

Derivative Controller

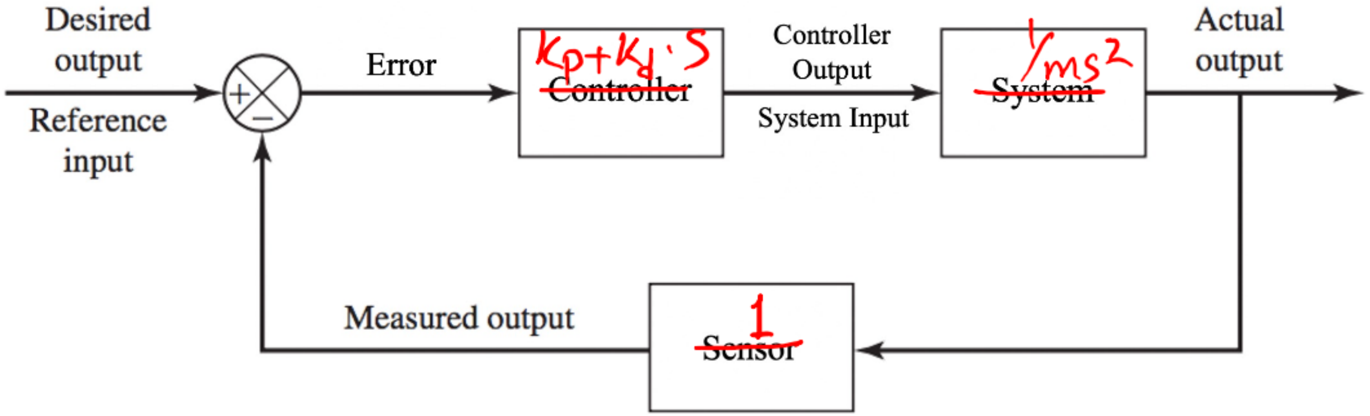
$F(S) = K_p \cdot E(S) + K_d \cdot S \cdot E(S)$



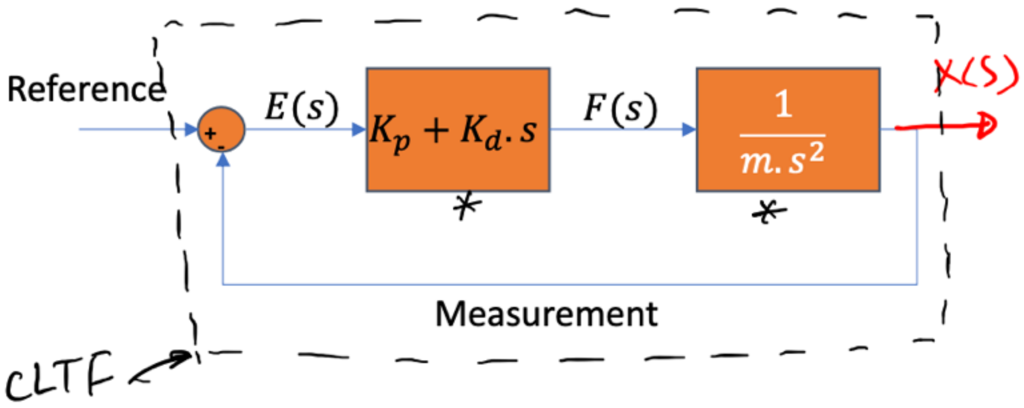
PD $\Rightarrow$ time domain: $F = K_p e + K_d \dot{e}$

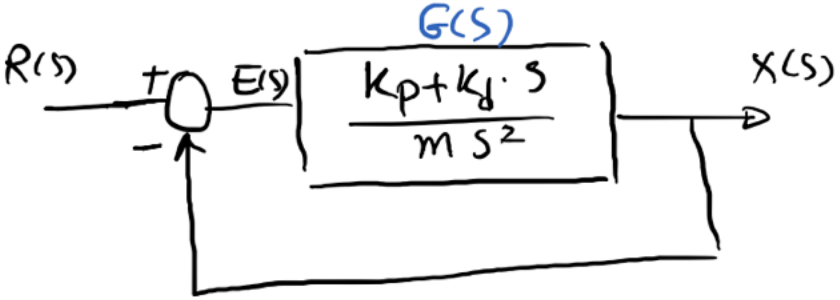
$\Rightarrow$ S-domain: $\dfrac{F(S)}{E(S)} = C(S) = K_p + K_d \cdot S$

$E(S) \rightarrow \boxed{K_p + K_d \cdot S} \rightarrow F(S)$



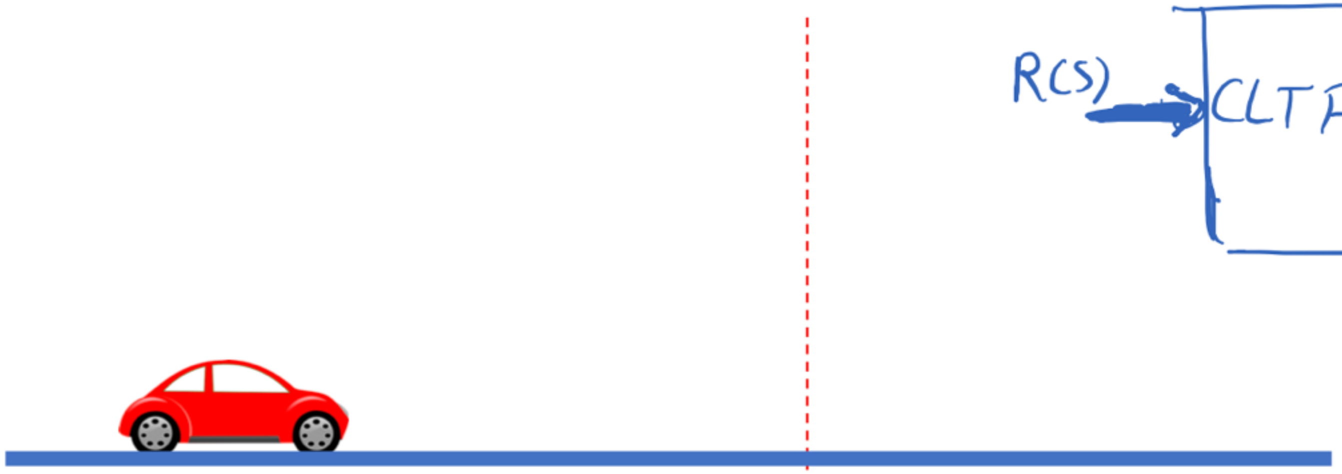
Derivative Controller





$$\Rightarrow CLTF = \frac{G(s)}{1 + G(s)}$$

$$R(s) \rightarrow \boxed{CLTF = \frac{k_d \cdot s + k_p}{ms^2 + k_d \cdot s + k_p}} \rightarrow X(s)$$

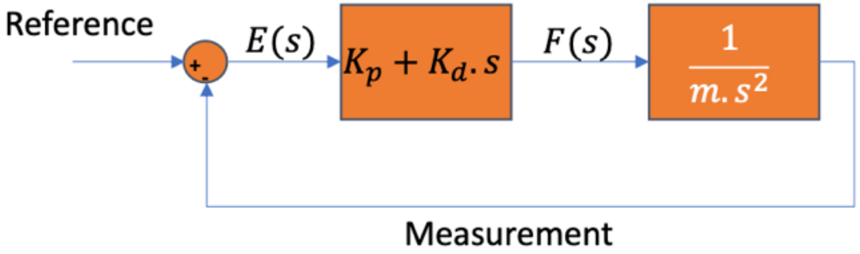


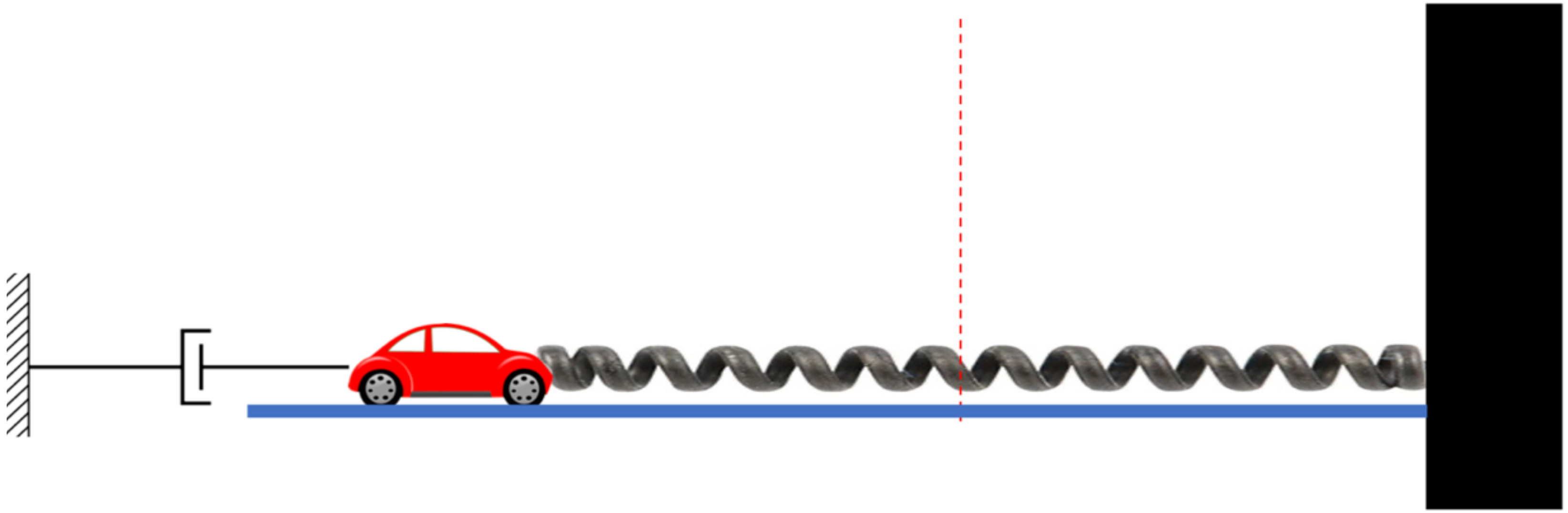
$m = 500$

$k_p > 0$

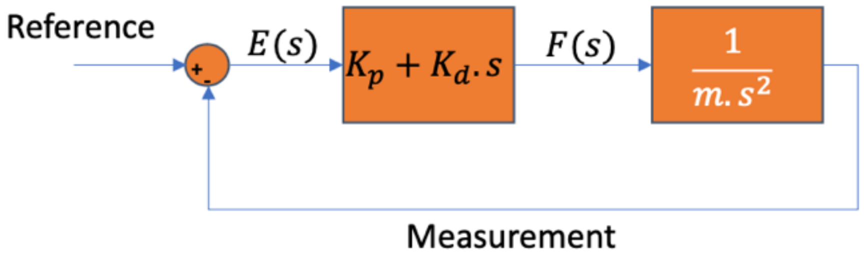
$k_d > 0$

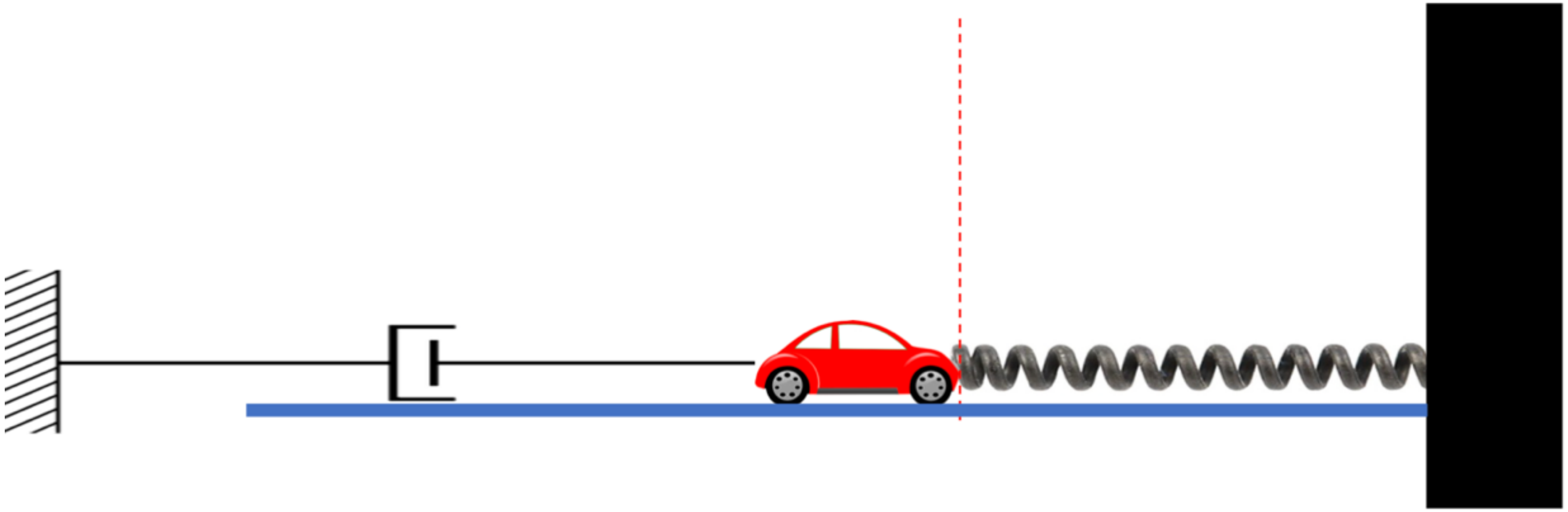
Derivative Controller





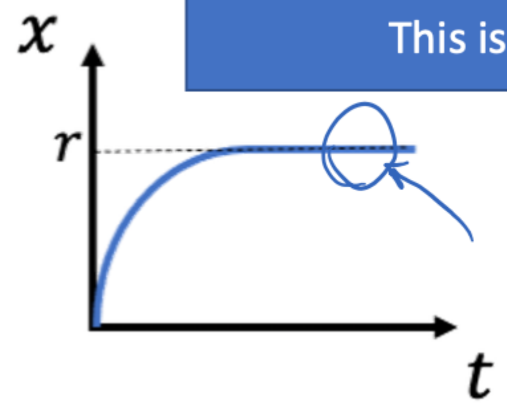
Derivative Controller



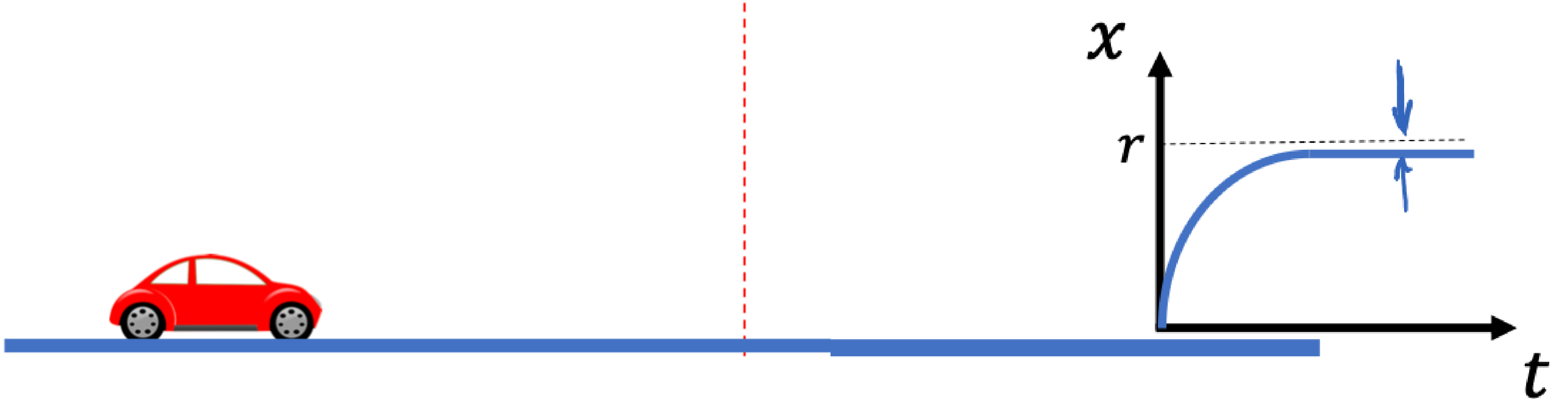


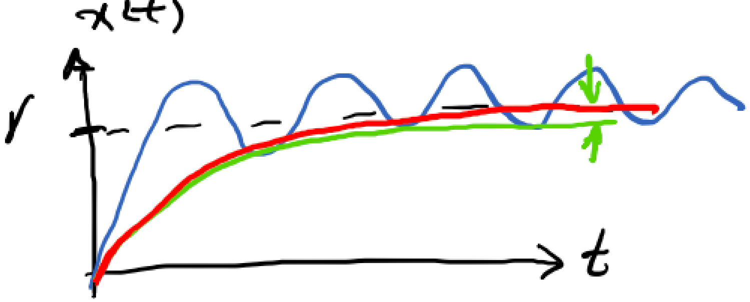
Remove the Steady-state Error

This is what we need.

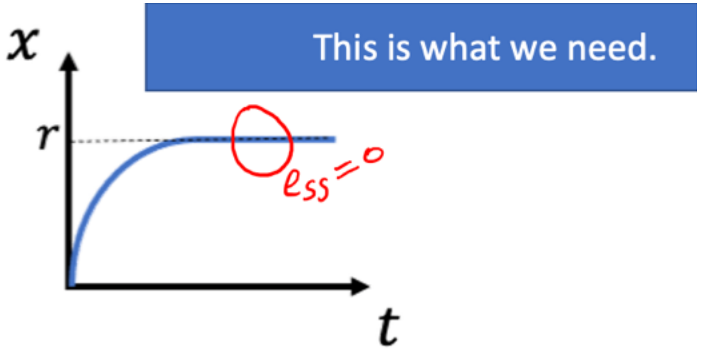


- Good, but it is not enough.
- We need to remove the steady state error.

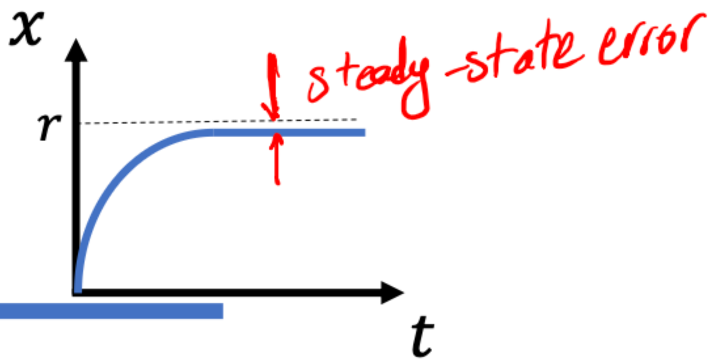




Integral Controller



- Good, but it is not enough.
- We need to remove the steady state error.



PID in time domain: $F = K_p + K_d\dot{e} + K_i\int e\,dt$

- For that we will add a term to the controller with the integral of the error

in s-domain:

$\mathcal{L} \to F(s) = K_p + K_d\, s\, E(s) + \frac{K_i E(s)}{s} \Rightarrow \frac{F(s)}{E(s)}$

$$C(s) = K_p + K_d . s + \frac{K_i}{s}$$

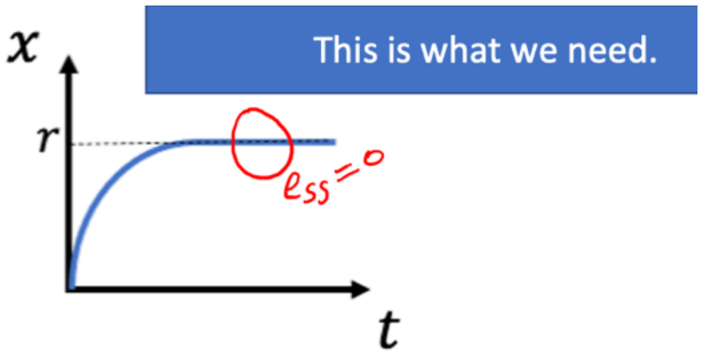
$C(s)$

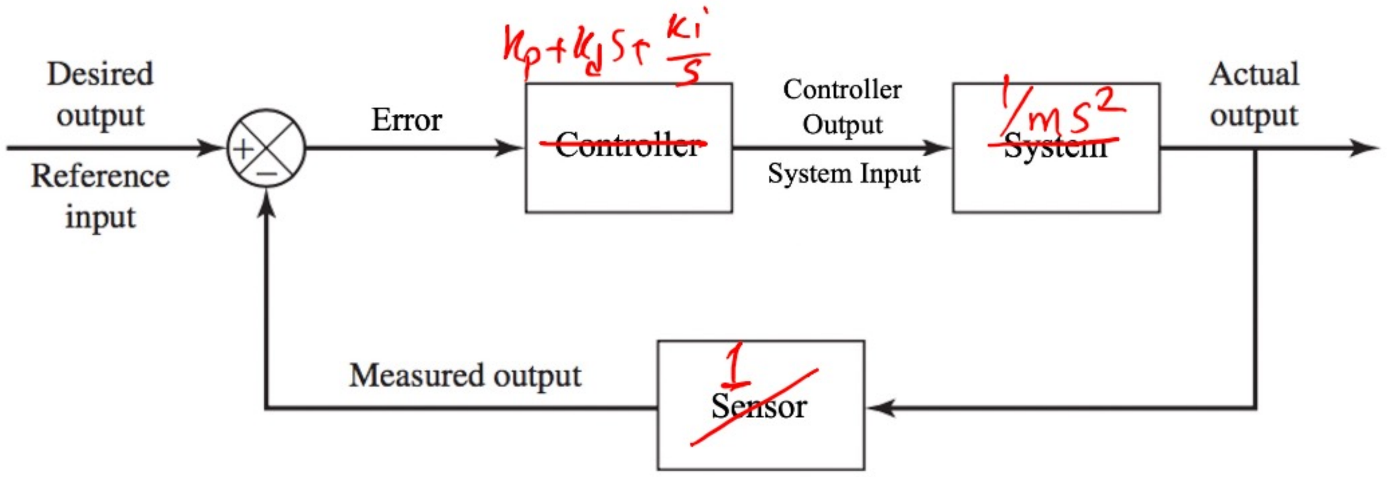
$E(s) \to \boxed{K_p + K_d s + K_i/s} \to F(s)$

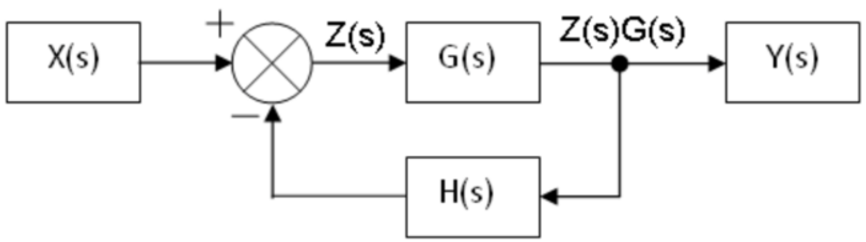
Integral Controller

- **Adding an Integral Controller (K_i/s) will help removing the Steady-state Error!**

Integral Controller

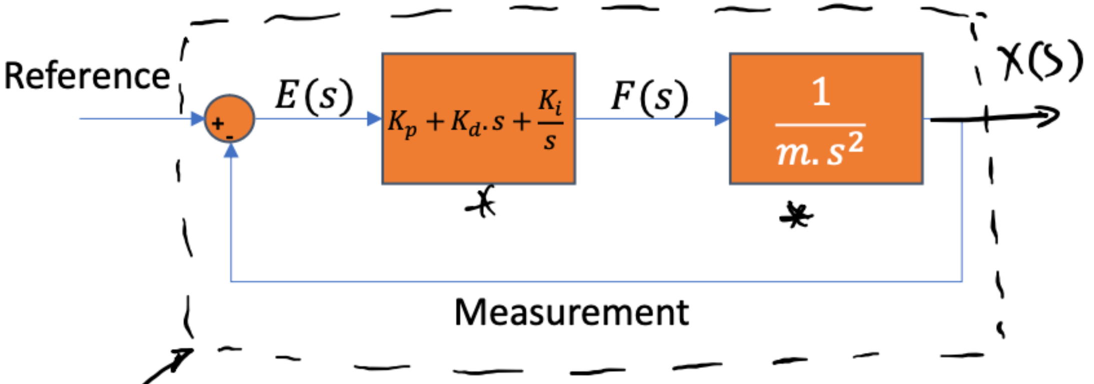




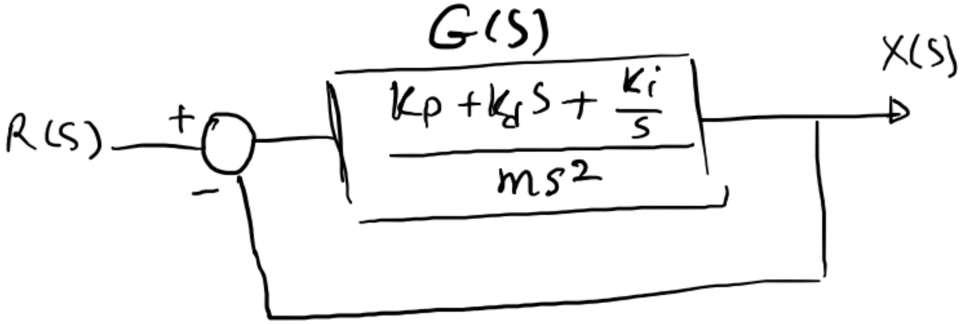


$$\frac{Y(s)}{X(s)} = \frac{G(s)}{1+G(s)H(s)}$$

Integral Controller

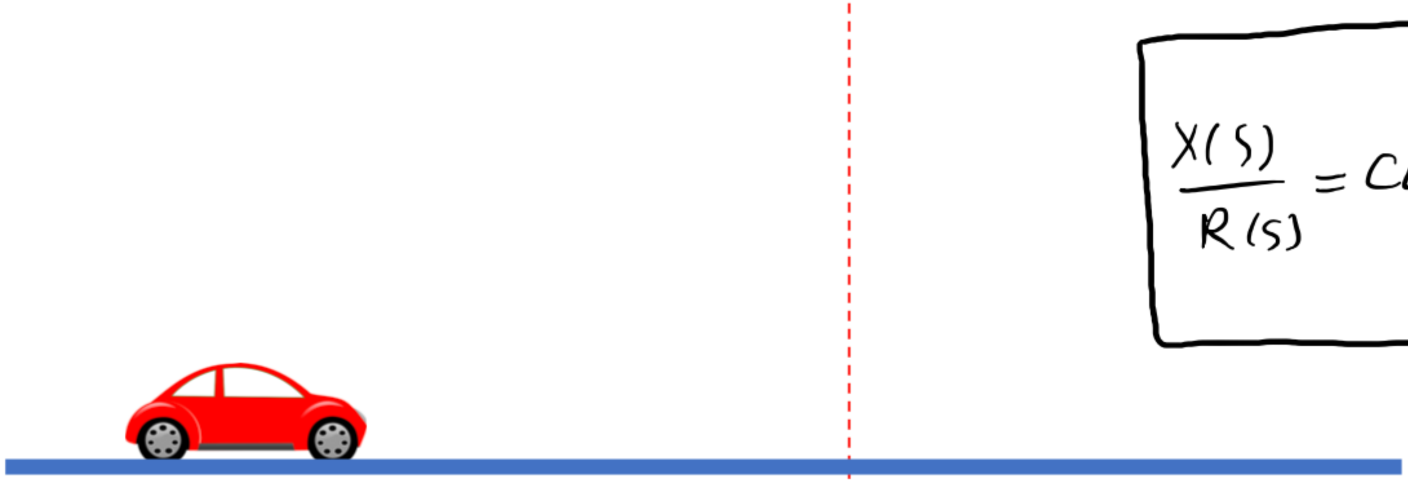


CLTF



$$\frac{X(S)}{R(S)} = \frac{G(S)}{1+G(S)}$$

$$\boxed{\frac{X(S)}{R(S)} = CLTF = \frac{K_d s^2 + K_p s + K_i}{m s^3 + K_d s^2 + K_p s + K_i}}$$



$m = 500$

$K_p, K_d, K_i > 0$

PID Controller - Gains

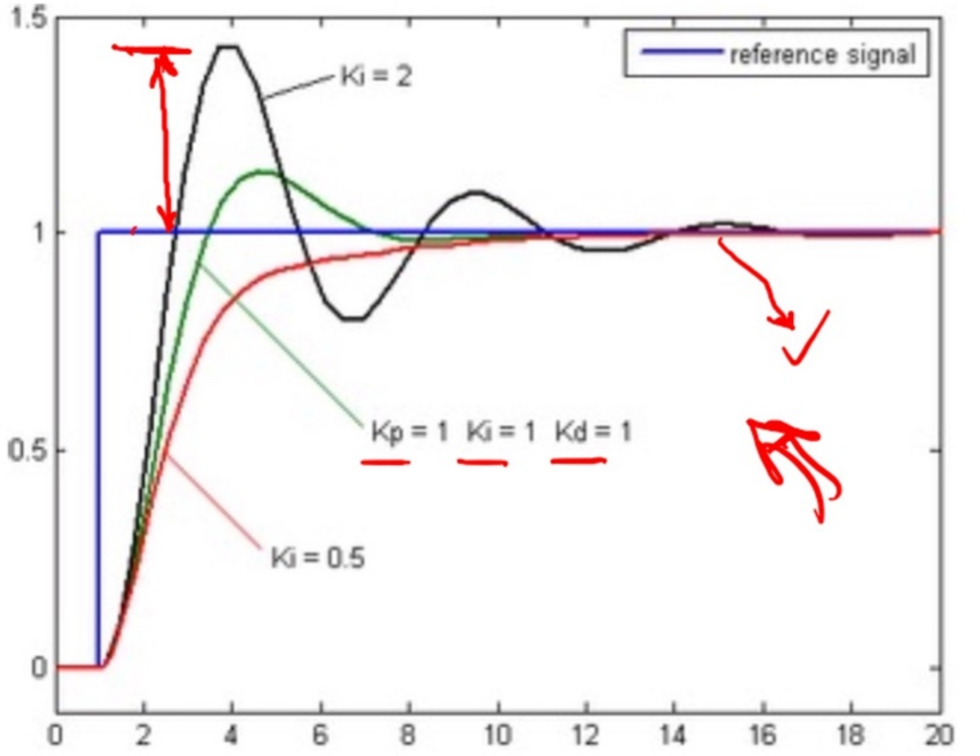
- This is PID controller in time domain (proportional, integral, derivative)

$$F = K_p e + K_d \dot{e} + K_i \int e$$

- This is PID controller in s-domain (complex frequency domain)

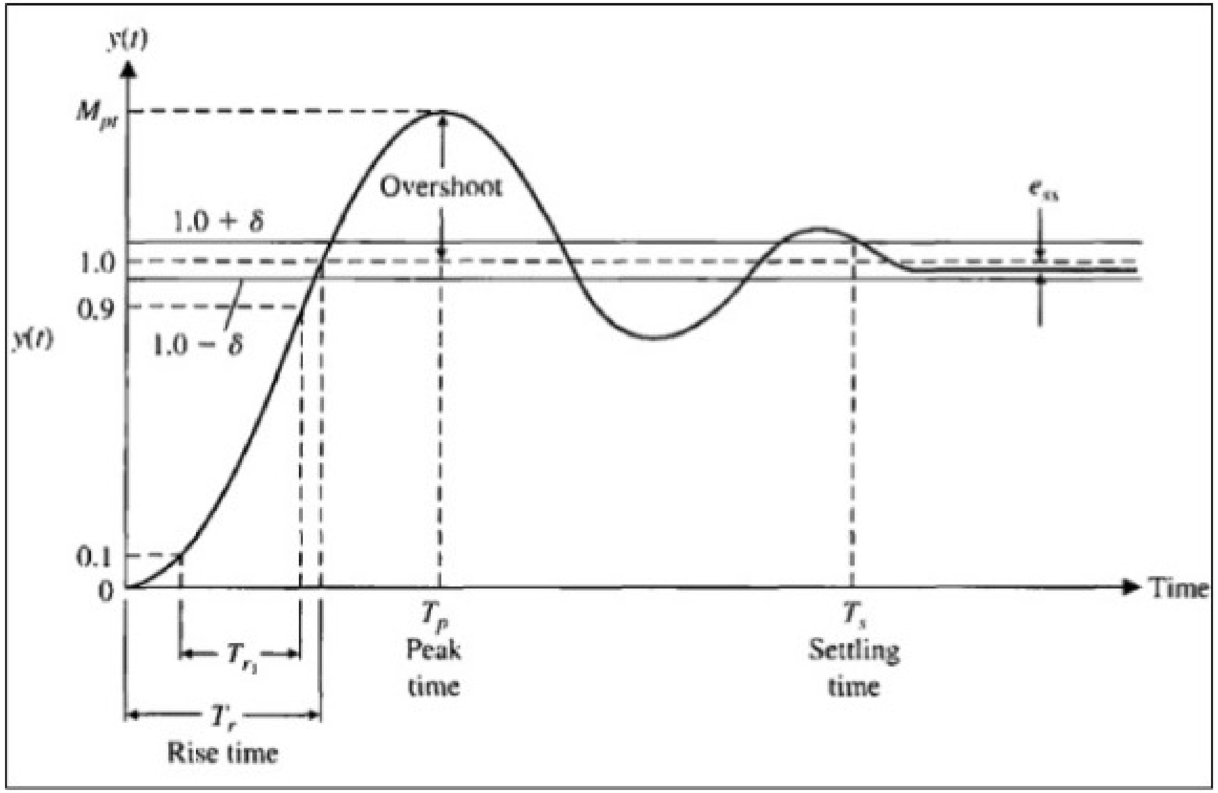
$$\frac{F(s)}{E(s)} = C(s) = K_p + K_d.s + \frac{K_i}{s}$$

- The $\boldsymbol{K_p}$, $\boldsymbol{K_i}$ and $\boldsymbol{K_d}$ are the **gains** of the controller
  - They are oftentimes tuned:
    - Trial error
    - There are also systematic procedures for tuning them



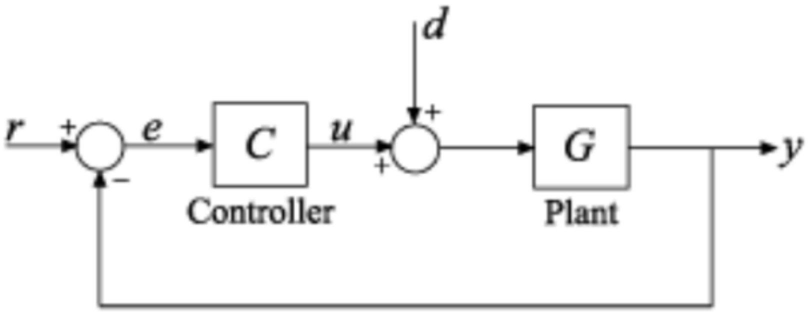
PID Controller - Gains effects

Parameter	Rise time	Overshoot	Settling time	Steady-state error	Stability
K_p	Decrease	Increase	Small change	Decrease	Degrade
K_i	Decrease	Increase	Increase	Eliminate	Degrade
K_d	Minor change	Decrease	Decrease	No effect in theory	Improve if K_d small



Second-order underdamped response specifications

Tune PID Controller to Favor Reference Tracking or Disturbance Rejection (PID Tuner) - in MATLAB



$$Plant = \frac{0.3}{s^2 + 0.1s}.$$

Design Initial PI Controller

Having an initial controller design provides a baseline against which you can compare results as you tune a PI controller. Create an initial PI controller design for the plant using PID tuning command `pidtune`.

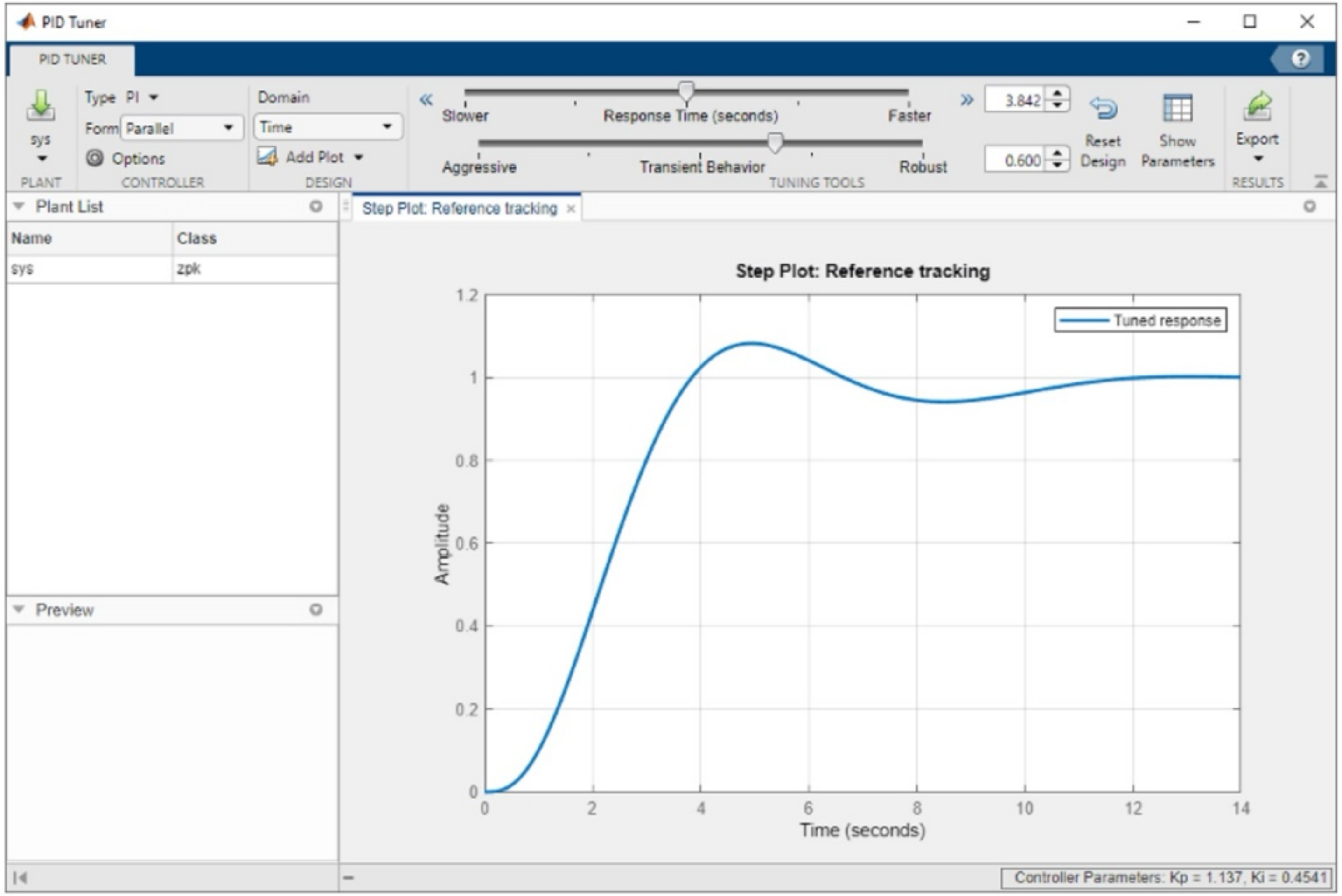
```
G = tf(0.3,[1,0.1,0]);    % plant model
C = pidtune(G,'PI');
```

Use the initial controller design to open **PID Tuner**.

```
pidTuner(G,C)
```


https://www.mathworks.com/help/control/getstart/tune-pid-controller-to-balance-tracking-and-disturbance-rejection.html

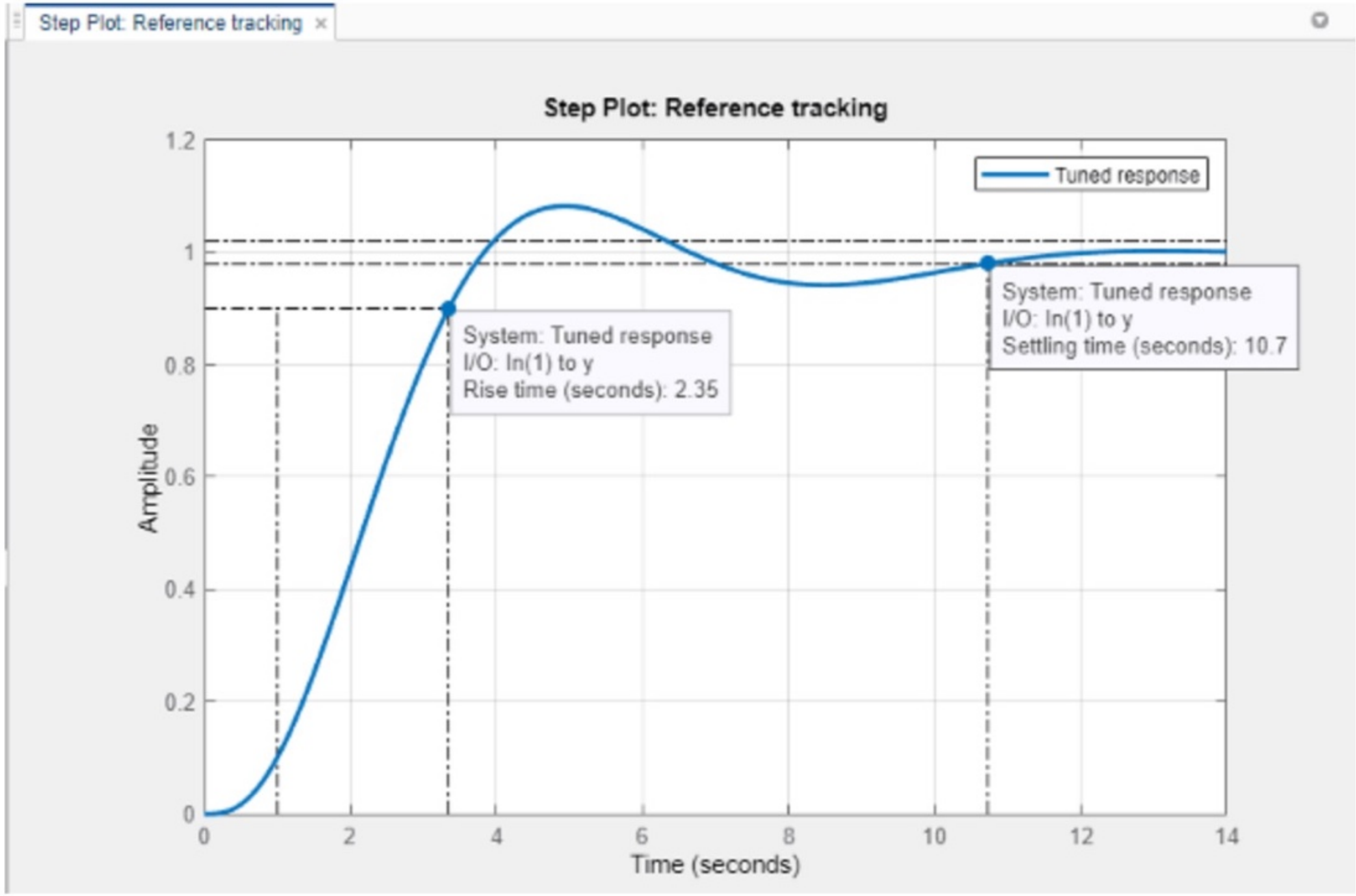
Tune PID Controller to Favor Reference Tracking or Disturbance Rejection (PID Tuner) - in MATLAB



Worcester Polytechnic Institute

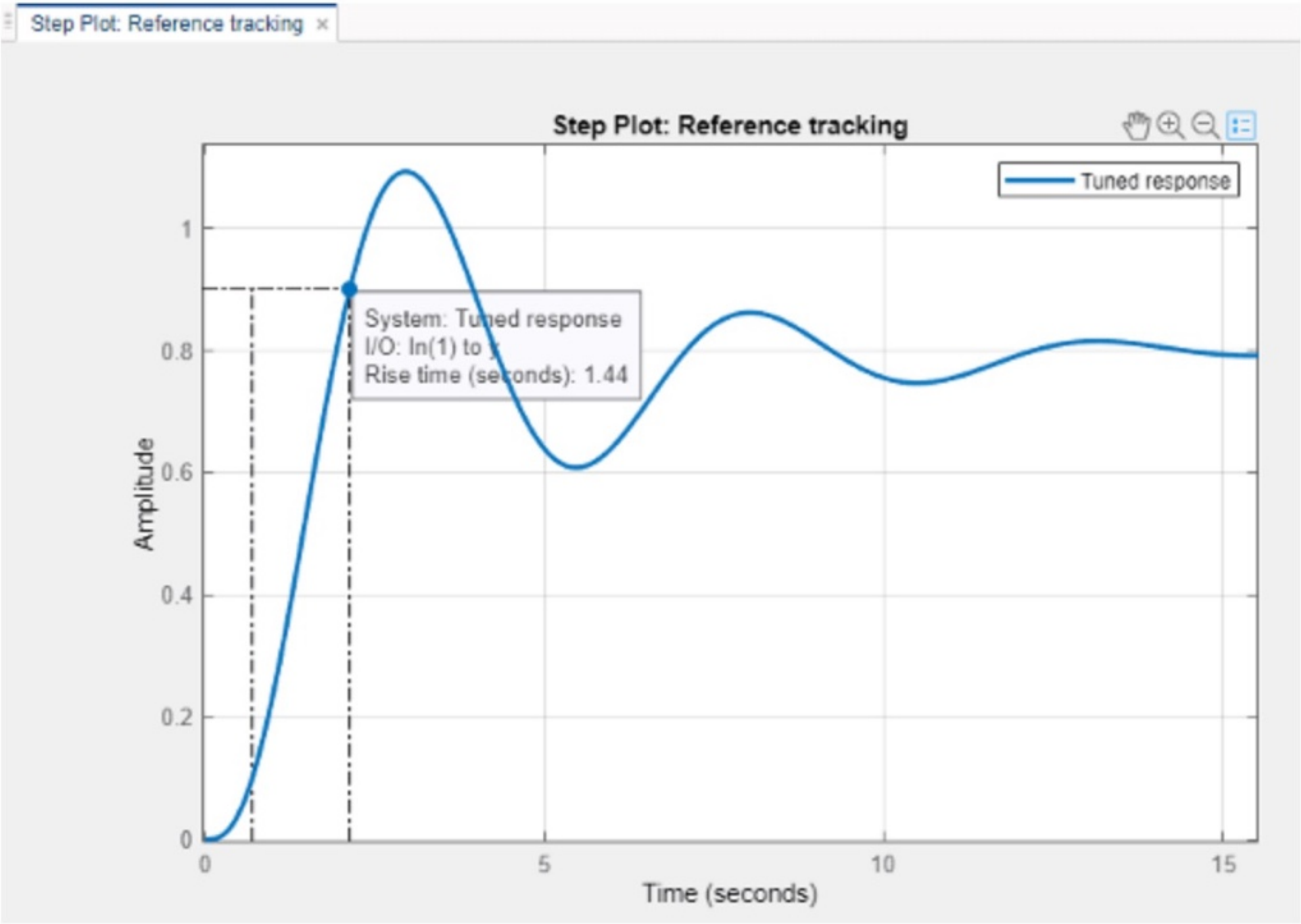
https://www.mathworks.com/help/control/getstart/tune-pid-controller-to-balance-tracking-and-disturbance-rejection.html

Tune PID Controller to Favor Reference Tracking or Disturbance Rejection (PID Tuner) - in MATLAB



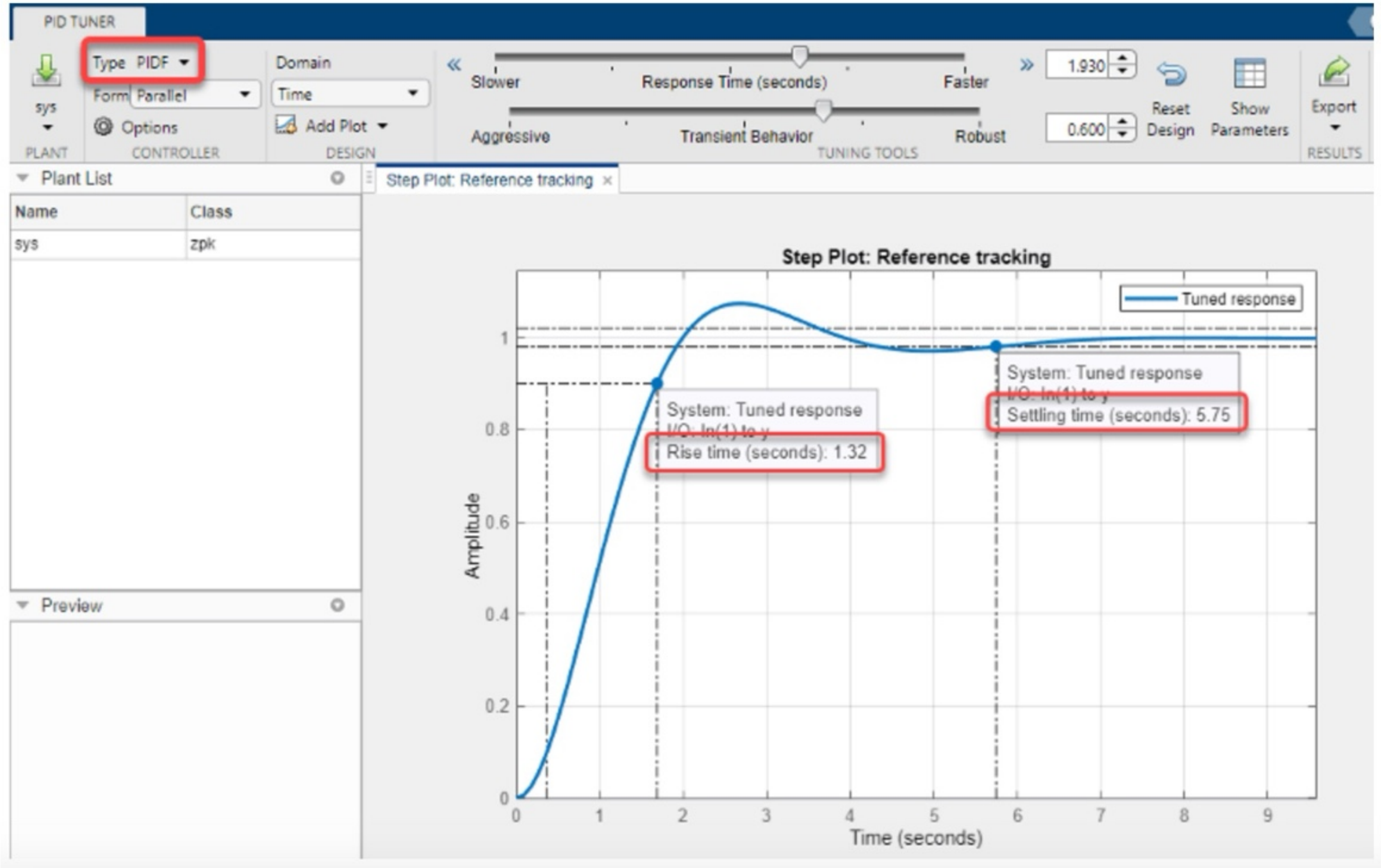
https://www.mathworks.com/help/control/getstart/tune-pid-controller-to-balance-tracking-and-disturbance-rejection.html

Tune PID Controller to Favor Reference Tracking or Disturbance Rejection (PID Tuner) - in MATLAB



https://www.mathworks.com/help/control/getstart/tune-pid-controller-to-balance-tracking-and-disturbance-rejection.html

Tune PID Controller to Favor Reference Tracking or Disturbance Rejection (PID Tuner) - in MATLAB



https://www.mathworks.com/help/control/getstart/tune-pid-controller-to-balance-tracking-and-disturbance-rejection.html

PID Tuner - Example

- Consider a system described by the equation of motion:

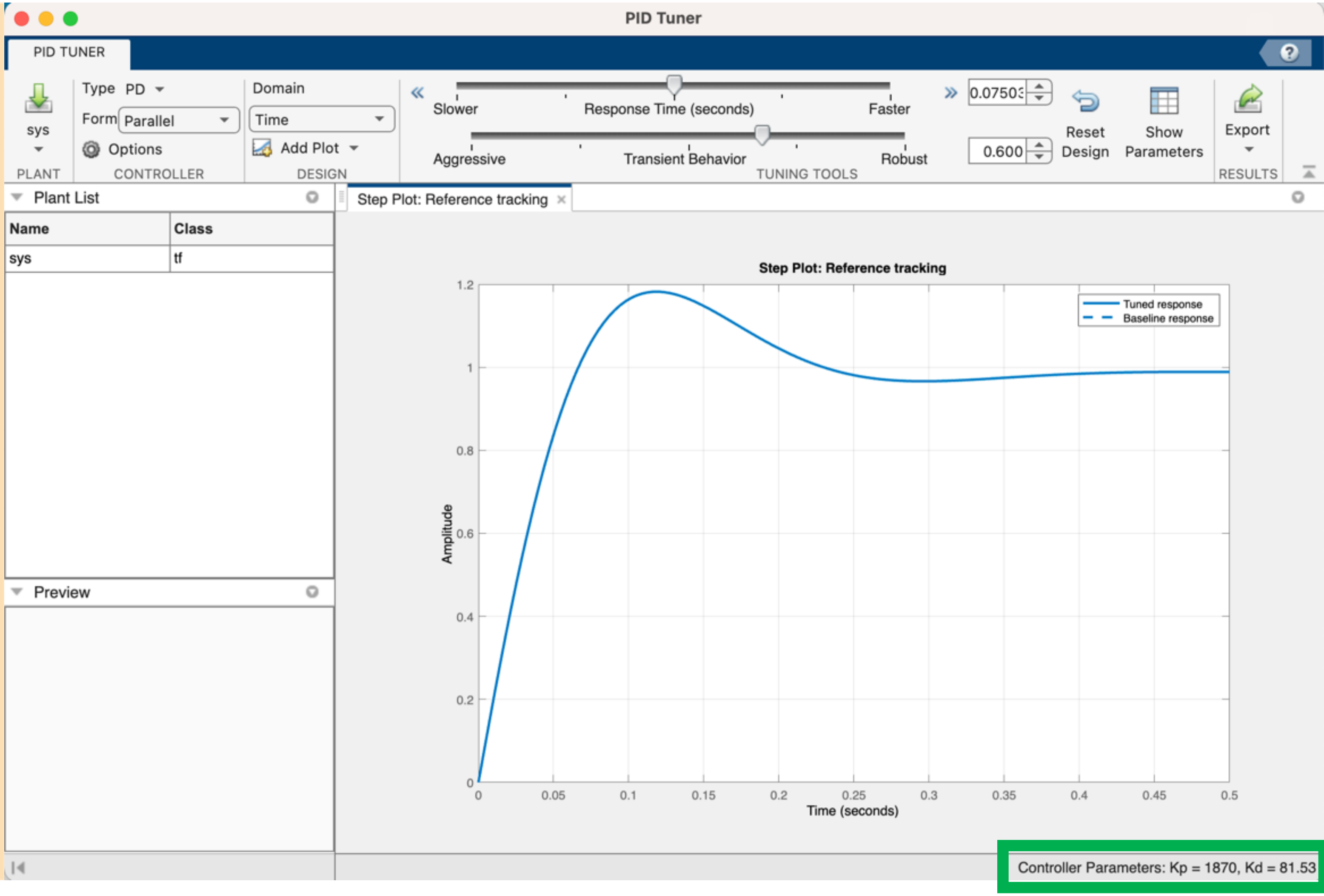
$$4\ddot{x} + 20\dot{x} + 25x = f$$

- Design a PD controller using the **pidTuner** function of **MATLAB**.

https://www.mathworks.com/help/control/getstart/tune-pid-controller-to-balance-tracking-and-disturbance-rejection.html

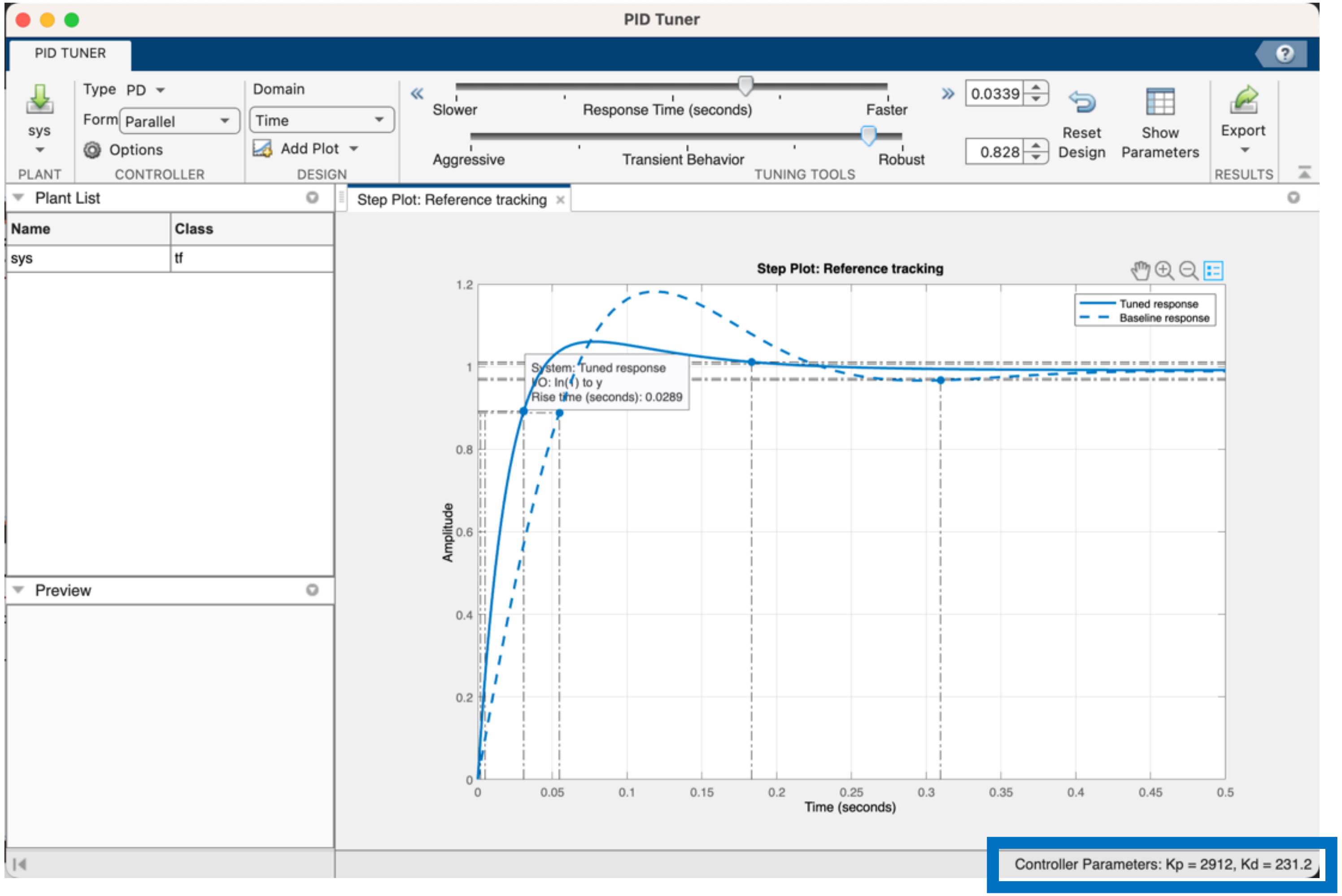
PID Tuner - Example - in MATLAB

```
>> sys = tf([1],[4,20,25])

sys =

          1
  -----------------
  4 s^2 + 20 s + 25

Continuous-time transfer function.

>> pd = pidtune(sys, 'pd')

pd =


  Kp + Kd * s

  with Kp = 1.87e+03, Kd = 81.5

Continuous-time PD controller in parallel form.

>> pidTuner(sys,pd)
```

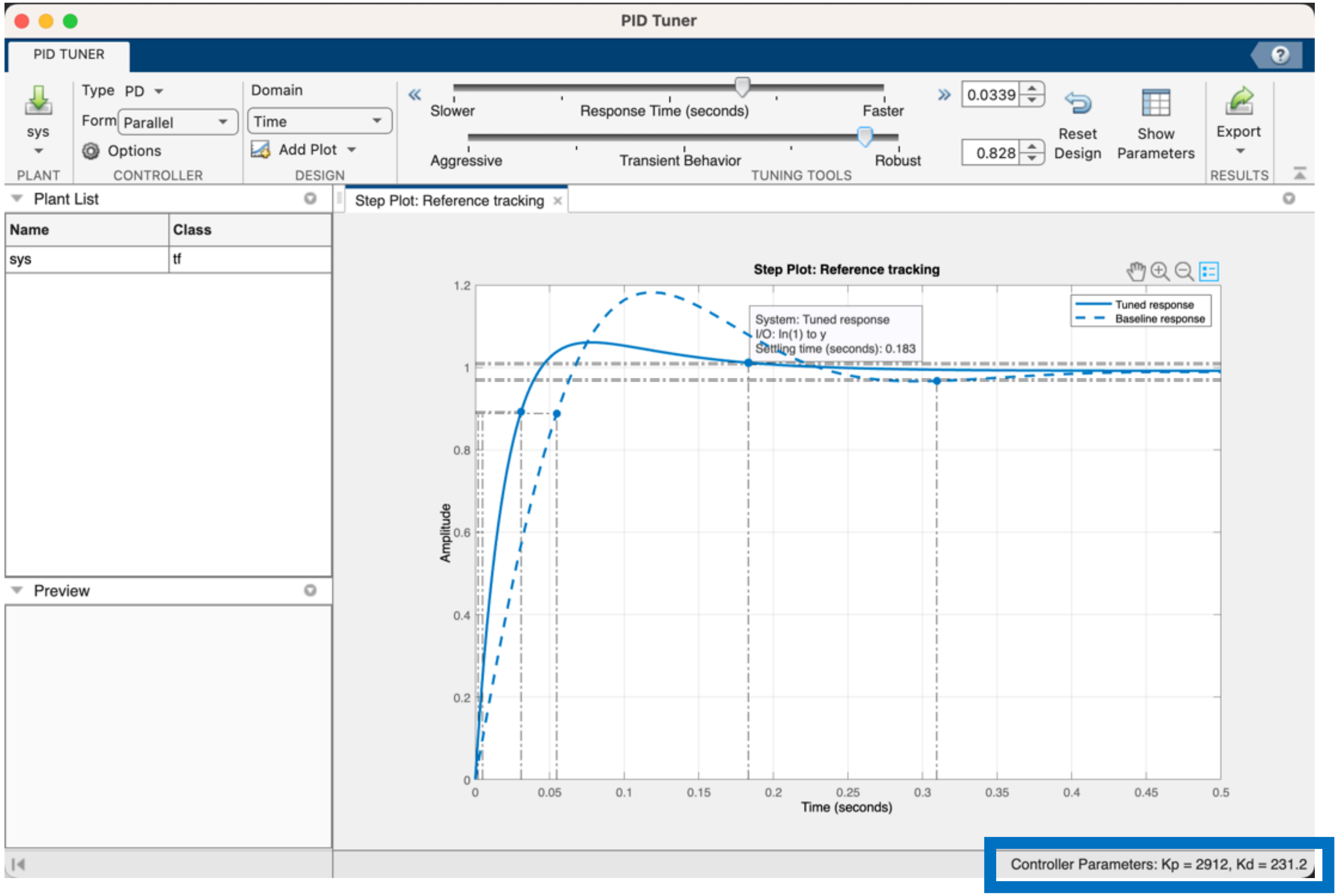


https://www.mathworks.com/help/control/getstart/tune-pid-controller-to-balance-tracking-and-disturbance-rejection.html

PID Tuner - Example - in MATLAB (Rise Time)

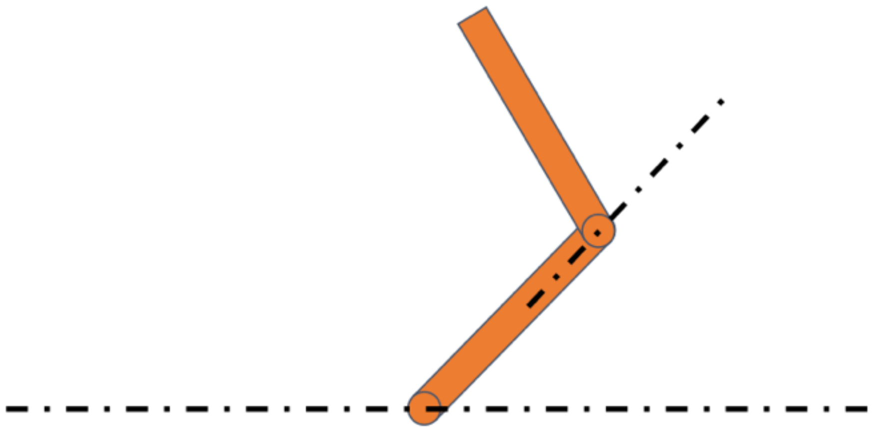


PID Tuner - Example - in MATLAB (Settling Time)

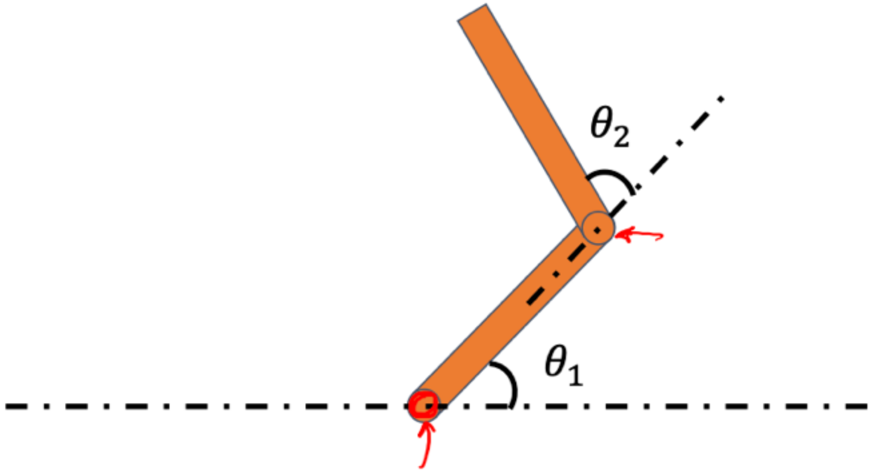


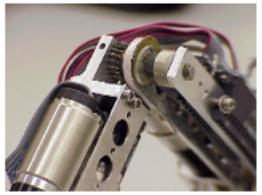
Two-link robotic manipulator

Two-link robotic manipulator



Two-link robotic manipulator - Joint positions



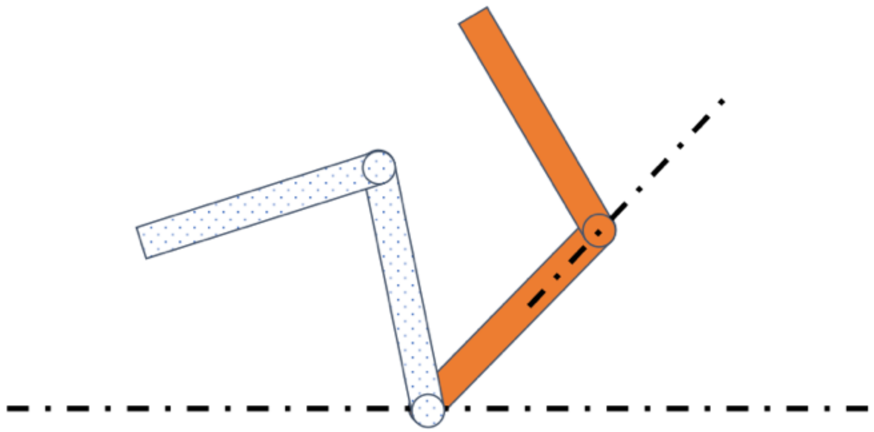


Encoders

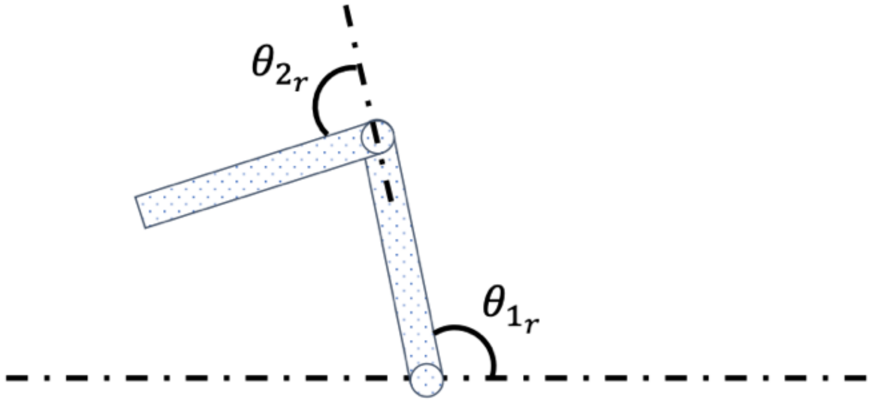
θ_{1_m}: Measured joint position

Sometimes only called θ_1

Two-link robotic manipulator - Desired position

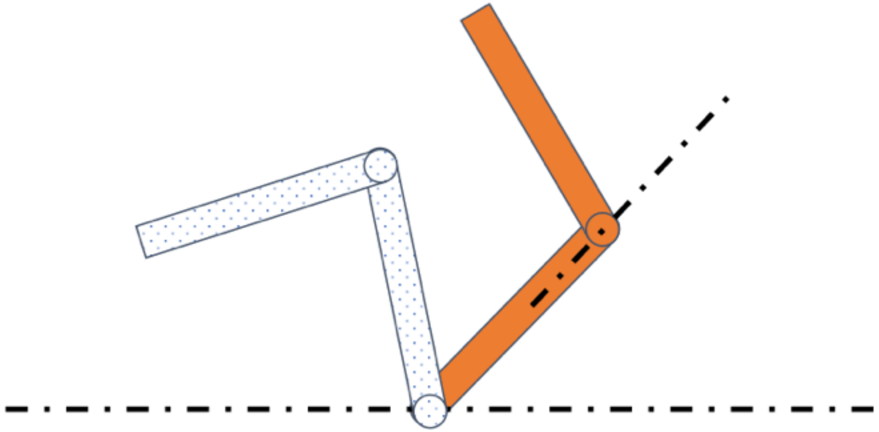


Two-link robotic manipulator - Joint Reference



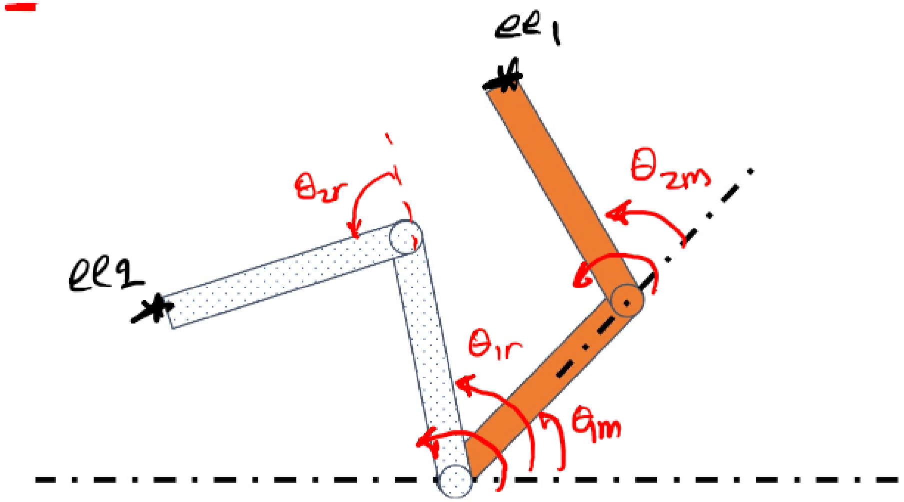
θ_{1_r}: Joint reference value

Position Control Problem



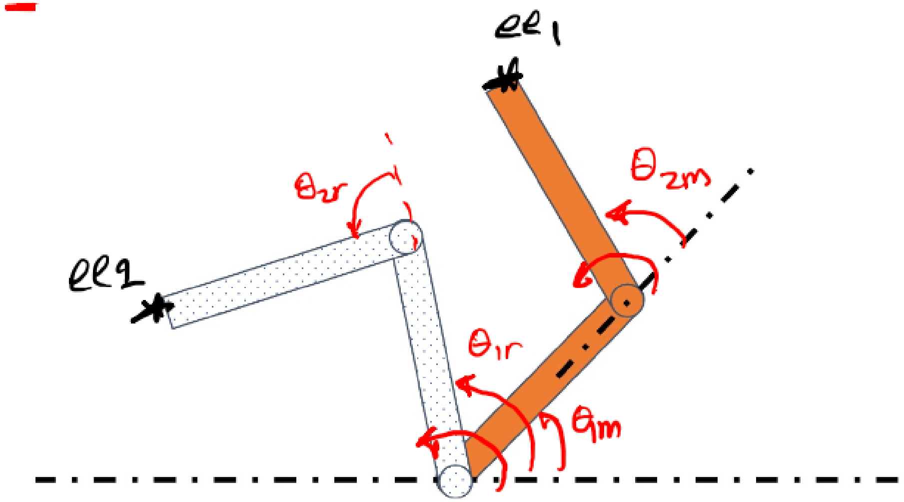
Position Control Problem

- We have a reference position value for the robot: θ_r
- We measure the current position of the robot: θ_m
- We can supply force: τ *or* u



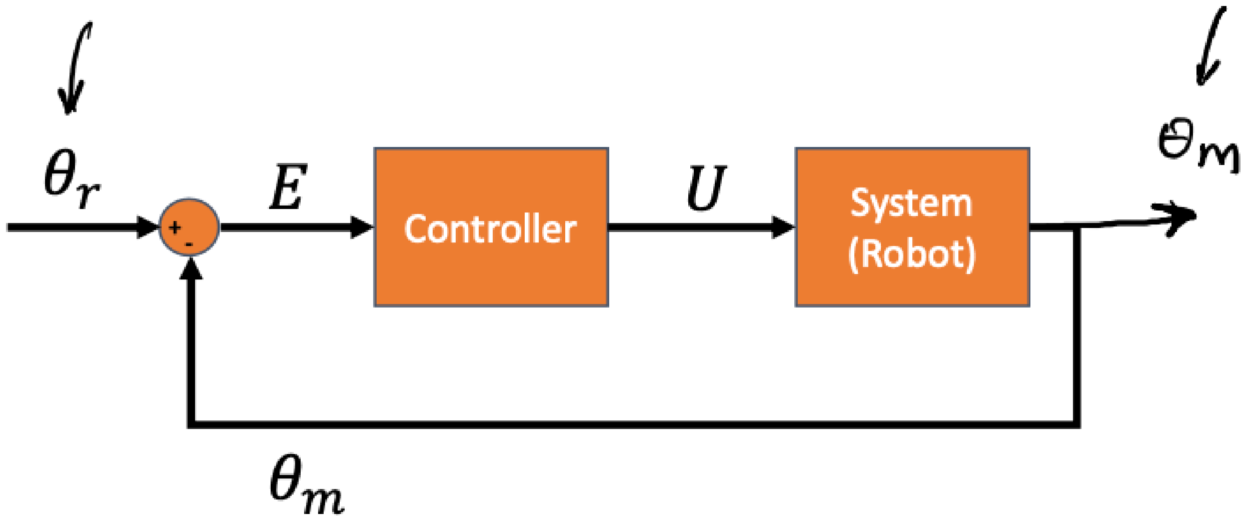
Position Control Problem

- We have a reference position value for the robot: θ_r
- We measure the current position of the robot: θ_m
- We can supply force: τ *or* u



How should I design u so that θ_m converges to θ_r?

Position Control Problem

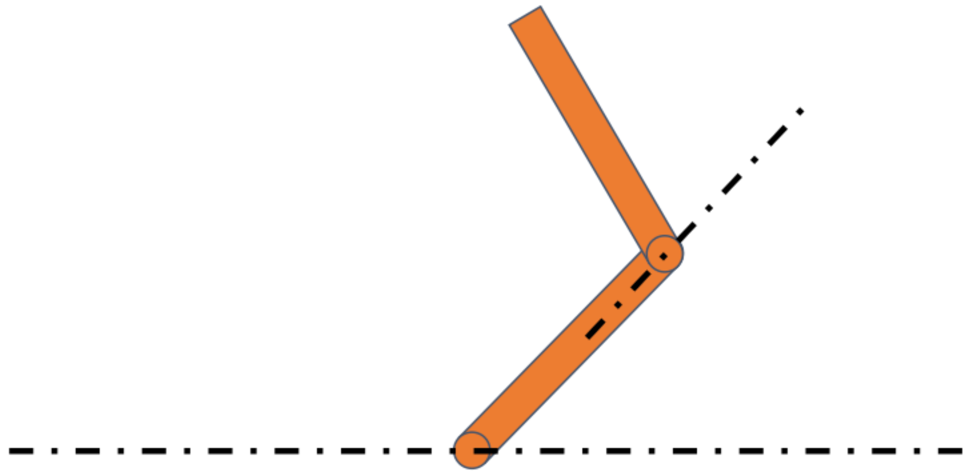


Position Control Problem

- What we have just mentioned is **joint space position control!**
- The end-effector position is given in **task space**.
- What do we do?

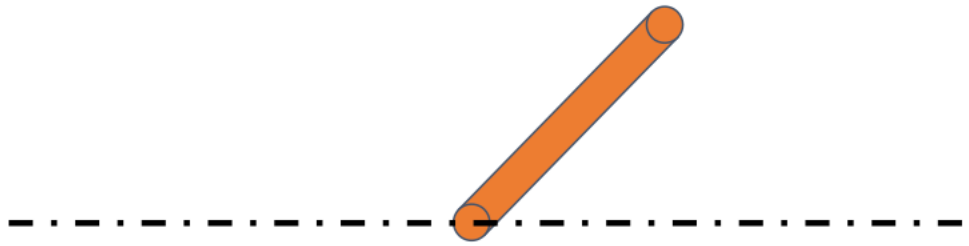
Position Control Problem

- Single input, single output (SISO) systems
  - Joints will be handled one by one



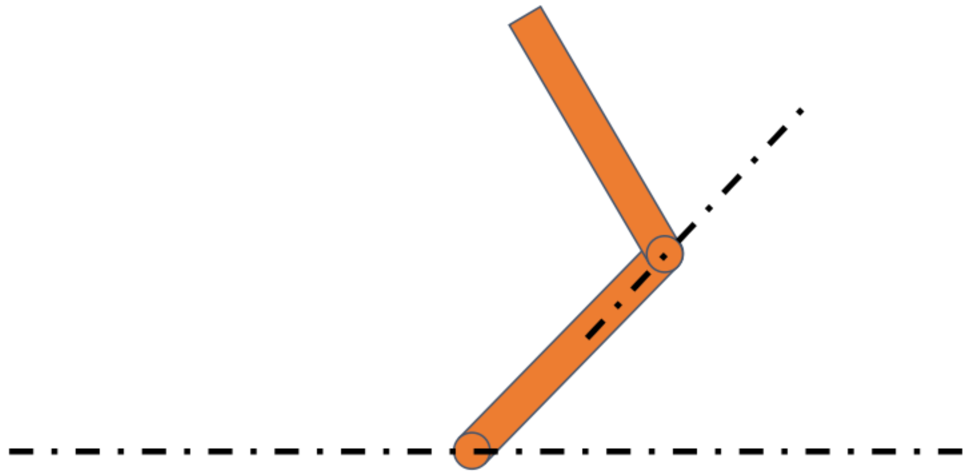
Position Control Problem

- Control the first link



Position Control Problem

- Control the second link



Position Control Problem

- So, we will learn about 1-link arm robot,
  and it can be repeated for other links as well !!!

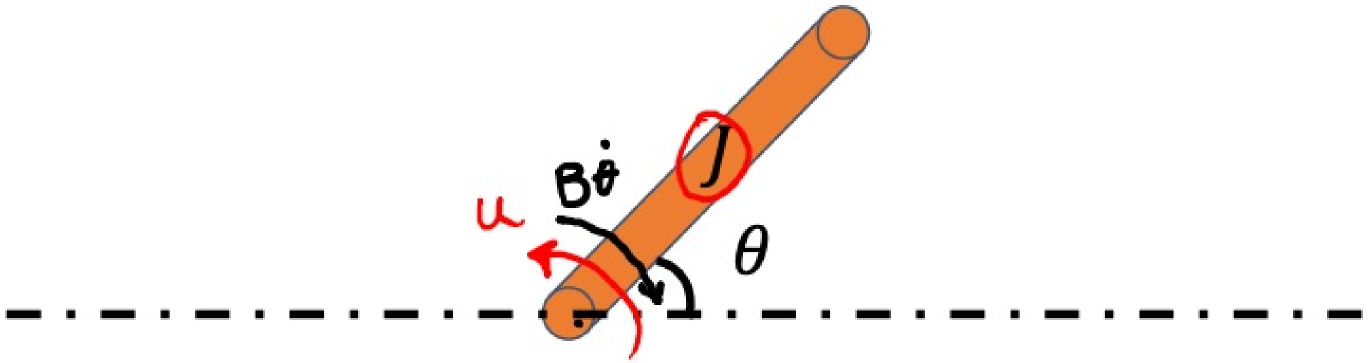
Steps for controlling a given system

1. Dynamic model
2. Transfer Function
3. Evaluation / Analysis
4. Add Controller (P, PD, PI, PID) and close the loop

1. Dynamic model

$$\tau_{net} = J\ddot{\theta}$$

torque inertia angular acceleration



$$u = J\ddot{\theta} + B\dot{\theta}$$

2. Transfer Function

$$u = J\ddot{\theta} + B\dot{\theta}$$

$$\mathcal{L} \rightarrow \boxed{U = Js^2\theta + Bs\theta}$$

$$U = Js^2\theta + Bs\theta$$

$$U = (Js^2 + Bs)\theta$$

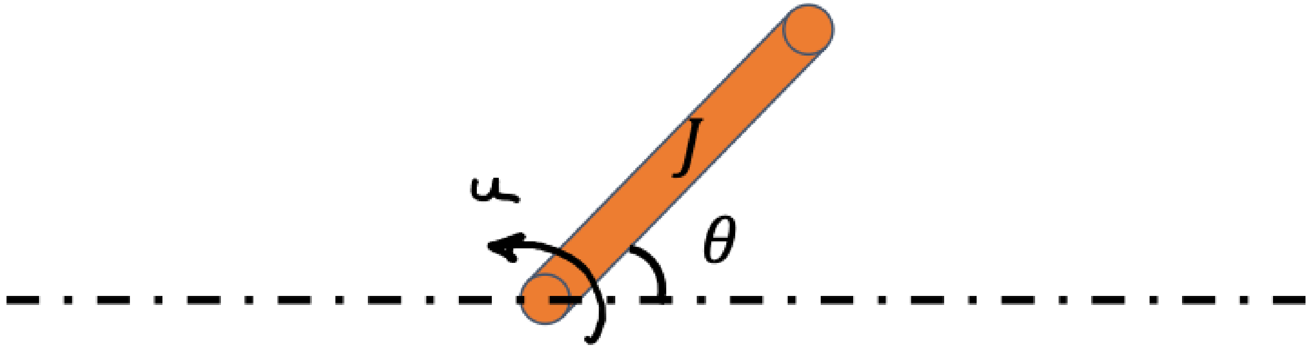
Output Signal ← θ

Input Signal ← U

$$\frac{\theta}{U} = \boxed{\frac{1}{(Js^2 + Bs)}}$$

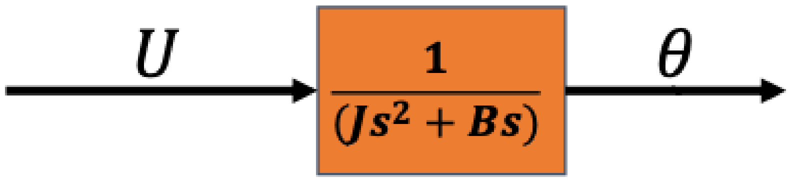
→ Transfer function of our system

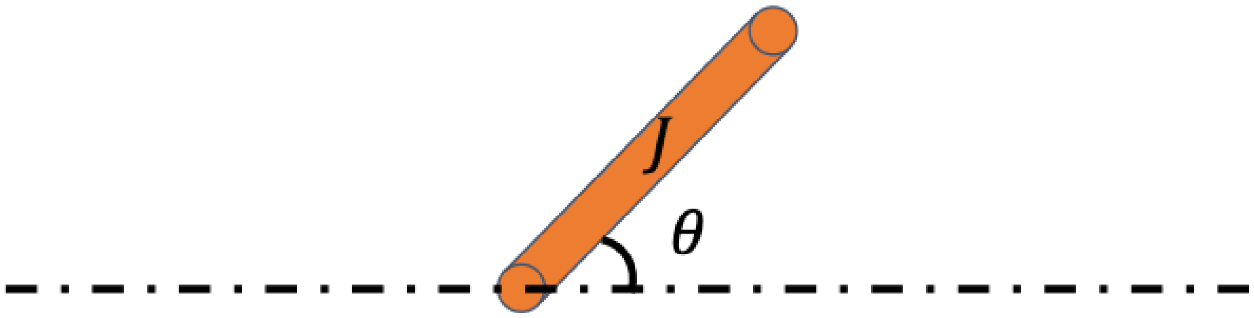
→ A transfer function transforms one signal to the other



3. Evaluation / Analysis

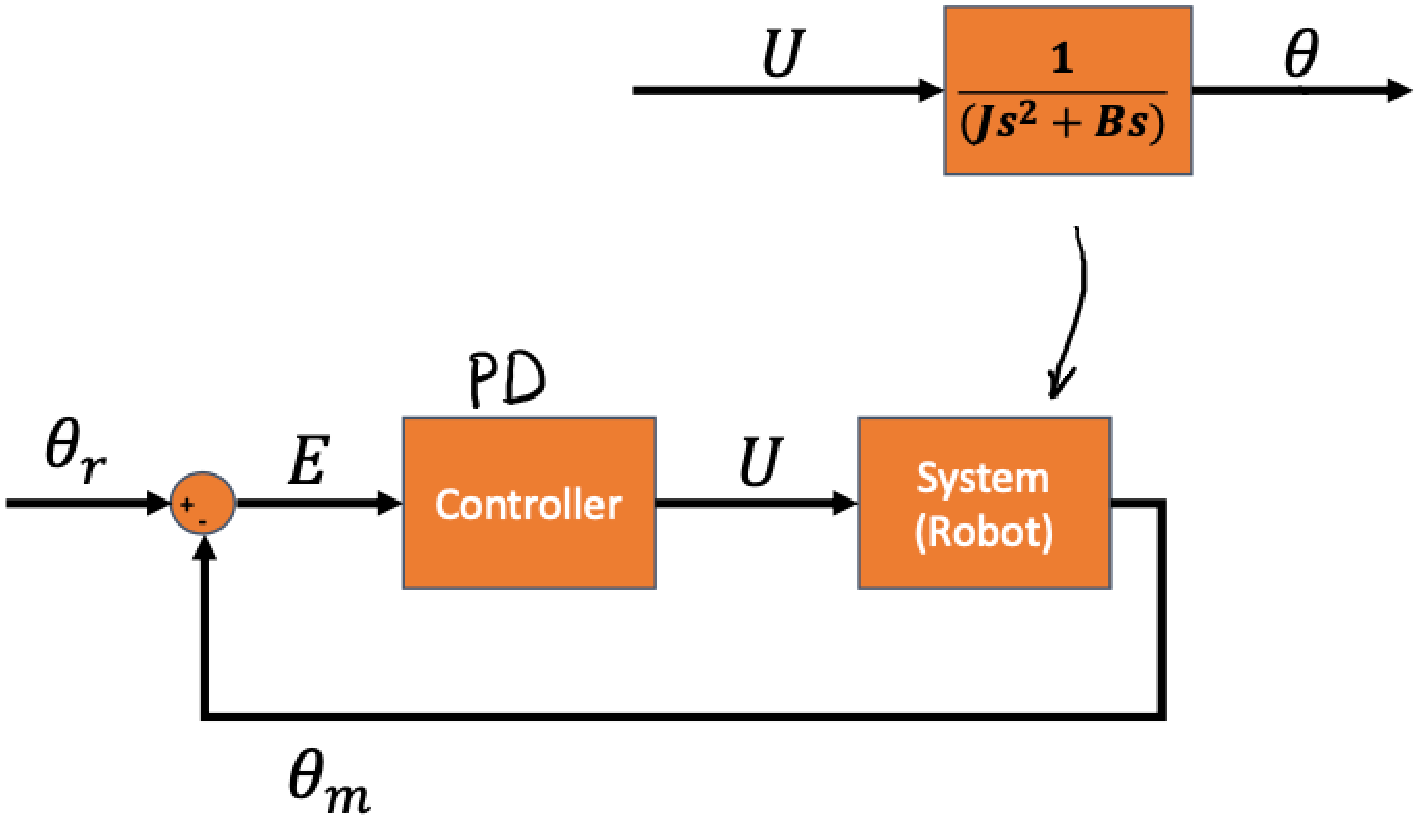
$$\frac{\theta}{U} = \frac{1}{(Js^2 + Bs)}$$



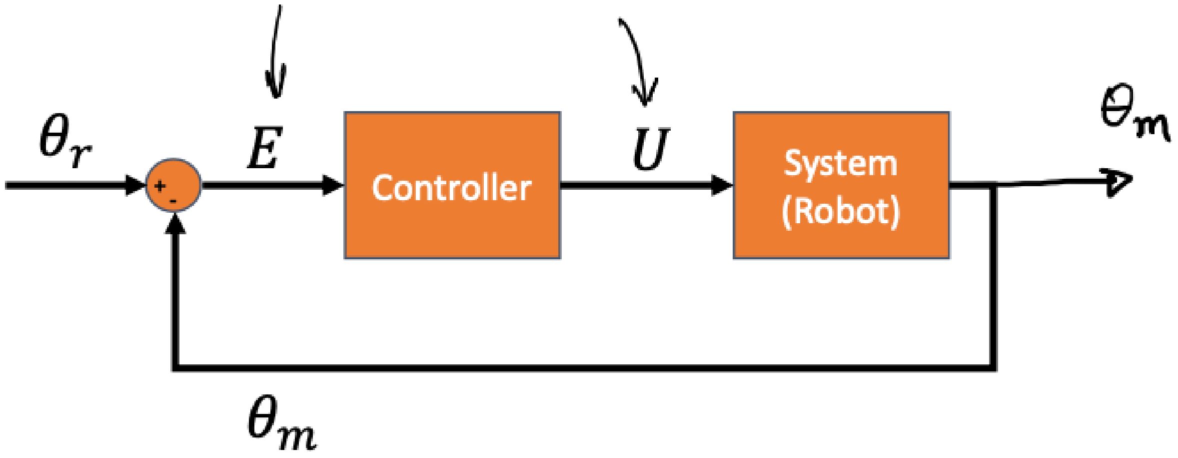


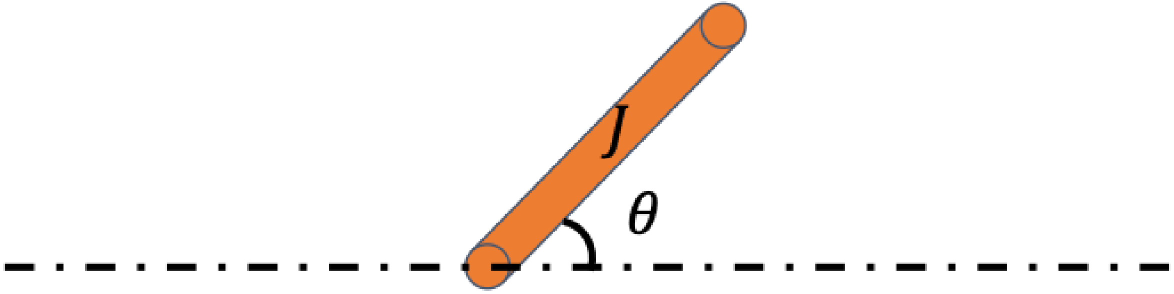
In Block Diagram

$$\frac{\theta}{U} = \frac{1}{(Js^2 + Bs)}$$



4. Add PD Controller and close the loop





PD Controller

- PD controller is

$$u(t) = K_p e(t) + K_d \dot{e}(t)$$

- In frequency domain

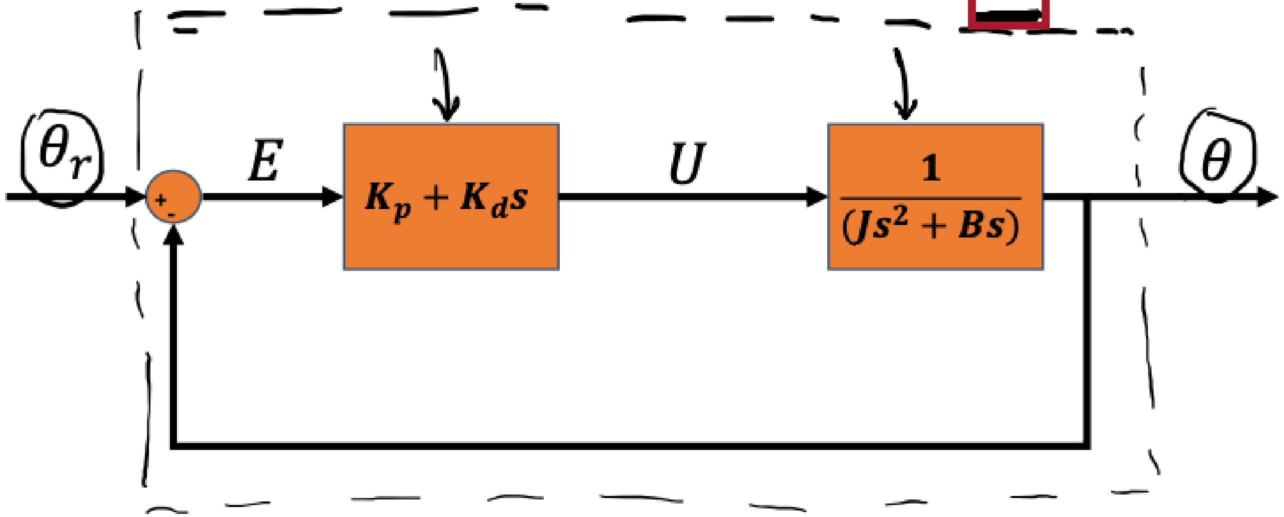
$$U(s) = K_p E(s) + K_d s E(s)$$

- Transfer function of our controller is

$$\frac{U(s)}{E(s)} = K_p + K_d s$$

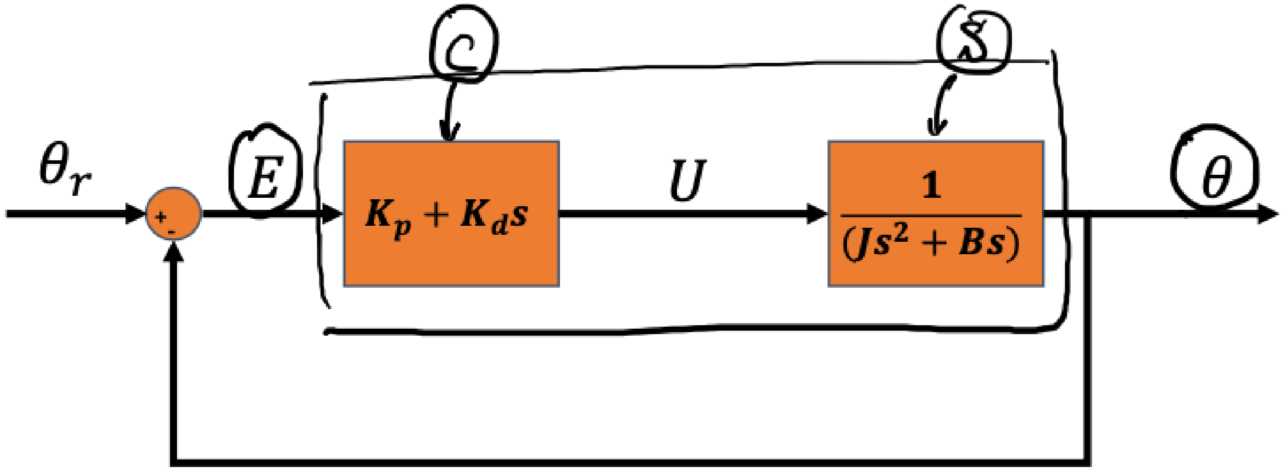
Closed-Loop Transfer Function

What we really want is to find $\boxed{\frac{\theta}{\theta_r}}$



Closed-Loop Transfer Function

Remember: Serial transforms get multiplied in frequency domain



$E \to \boxed{C} \to U$: $\frac{U}{E} = C \Rightarrow U = \underbrace{EC}$

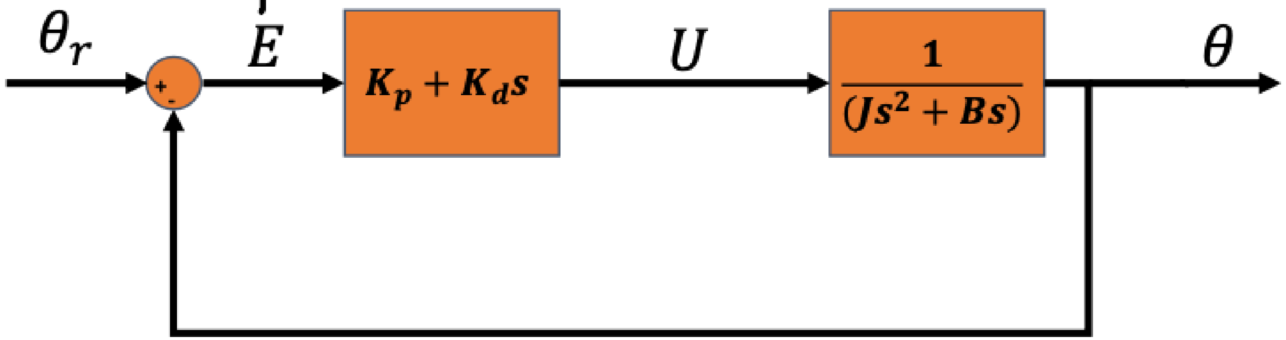
$U \to \boxed{S} \to \theta \Rightarrow \frac{\theta}{U} = S \Rightarrow \theta = US \Rightarrow \theta = ECS \Rightarrow \frac{\theta}{E} = CS$

$E \to \boxed{CS} \to \theta$

Closed-Loop Transfer Function

Remember: Serial transforms get multiplied in frequency domain

$E = \theta_r - \theta$



$$\boxed{\frac{\theta}{\theta_r} = \frac{G}{1+G}}$$

$$E(s)\frac{(K_p + K_d s)}{Js^2 + Bs} = \theta$$

$$\Rightarrow G = \frac{\theta}{E} = \frac{K_p + K_d \cdot s}{Js^2 + Bs} = G$$

Closed-Loop Transfer Function

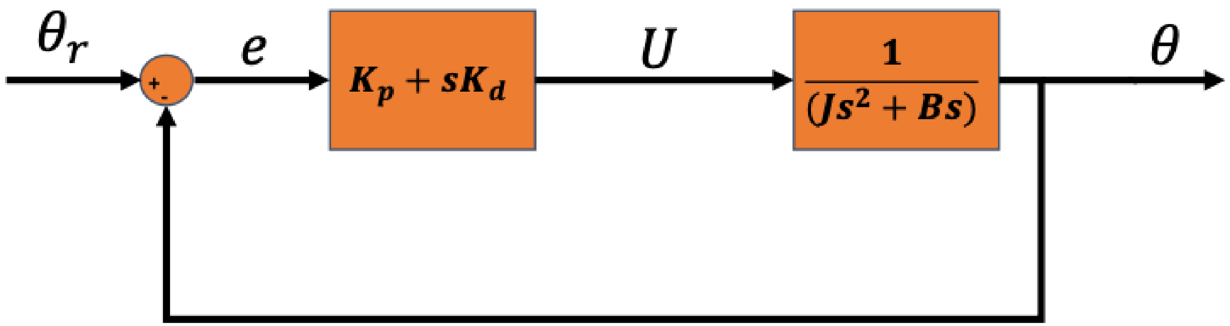
$$E(s)\frac{(K_p + K_d s)}{Js^2 + Bs} = \theta(s)$$

$$E(s) = \theta_r - \theta$$

$$\theta_r - \theta = \frac{Js^2 + Bs}{(K_p + K_d s)}\theta$$

$$\theta_r = \left(1 + \frac{Js^2 + Bs}{(K_p + K_d s)}\right)\theta$$

$$\boxed{\frac{\theta}{\theta_r} = \left(\frac{(K_p + K_d s)}{Js^2 + (B + K_d)s + K_p}\right)}$$



… end of Lecture 13