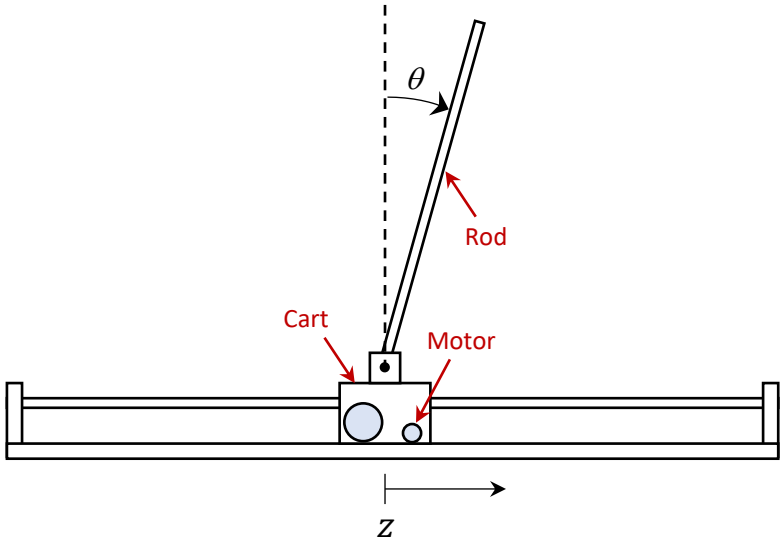
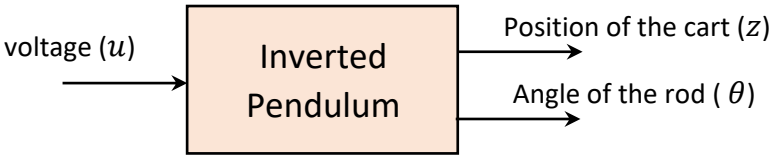
Course: Methods and algorithms for advanced process control
By Oscar Mauricio Agudelo, PhD
KU Leuven-ESAT-STADIUS

Design of a stabilizing controller for the Inverted Pendulum



Control goal: Keep the rod upright by manipulating the voltage applied to the DC motor attached to the Cart.



The linear state space model of the system is given by

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}u$$
$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x}$$

with

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -4.5231 & -16.8835 & 0 \\ 0 & 46.9609 & 55.3557 & 0 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 0 \\ 0 \\ 3.7778 \\ -12.3862 \end{bmatrix} \text{ and } \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

where $\boldsymbol{x} = [z \quad \theta \quad \dot{z} \quad \dot{\theta}]^T$ is the state vector, z is position of the cart in [m], θ is the angle of the rod in [rad], $\dot{z}$ is the velocity of the cart in [m/s], $\dot{\theta}$ is the angular velocity of the rod in [rad/s] and u is the voltage in [v] applied to DC motor attached to the cart. This model was obtained after linearizing the nonlinear differential equations of the system around the origin. The linearization is valid for $-15 < \theta < 15$ degrees.

Note: the Matab script **"ip_example.m"** contains the code for performing all the computations of this exercise.

The eigenvalues of $\boldsymbol{A}$ (and therefore the poles of the system) are the roots of its characteristic polynomial, $\det(s\boldsymbol{I} - \boldsymbol{A}) = 0$. From the following Matlab code,

Matlab Code

```
A = [ 0          0         1          0
      0          0         0          1
      0    -4.5231    -16.8835     0
      0     46.9609     55.3557     0];
B = [ 0
      0
      3.7778
      -12.3862];
C = [1  0  0  0
     0  1  0  0];
D = [0
     0];

eig(A)
```

we can get the poles of the system, which are:

Pole	Stable/Unstable
0	Marginally stable (integrator)
-17.8100	Stable
6.0017	Unstable
-5.0752	Stable

Clearly the system is unstable due to the pole at 6.0017.

Before embarking in the design of a state feedback controller via pole placement, it is important to check if we can arbitrary place the closed-loop poles wherever we want. Said in other words, we must check if the system is controllable or not. The controllability matrix of the inverted pendulum would be given by

$$\mathcal{C} \in \mathbb{R}^{4\times4} = [\boldsymbol{B} \quad \boldsymbol{AB} \quad \boldsymbol{A}^2\boldsymbol{B} \quad \boldsymbol{A}^3\boldsymbol{B}].$$

The rank of this matrix is 4, which is equal to the order of the system and therefore the system is completely controllable.

Matlab code

```
CO = ctrb(A,B)
rank(CO)
```

Now, it is time to pick the desired locations of the closed-loop poles. So, let's use what we already know about standard second order systems to select the dominant poles (the stable poles closest to the imaginary axis in the s-plane).

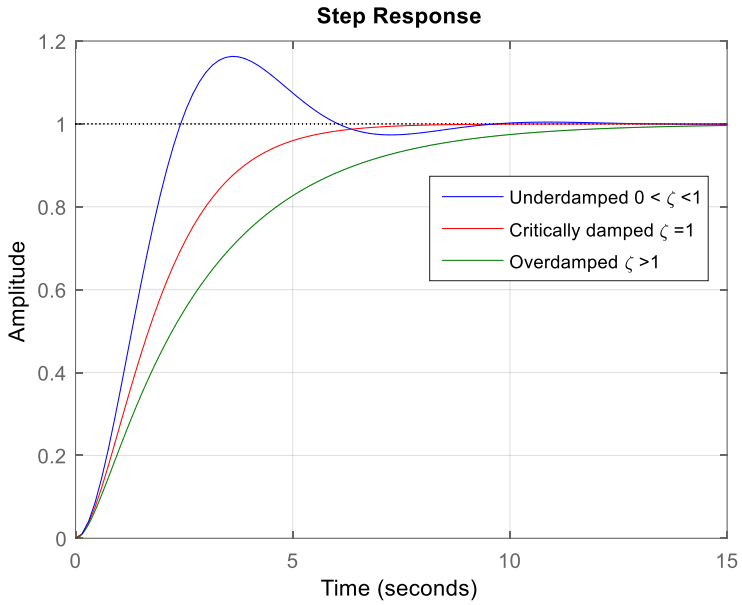
Recap

Standard form of a second order system:

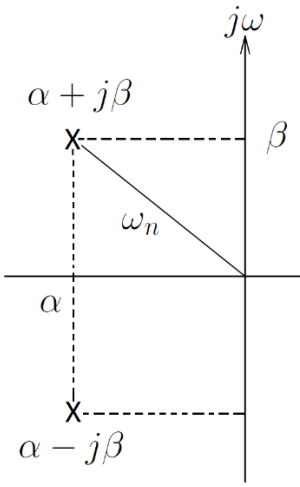
$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

where ω_n is the natural frequency in rad/s and ζ is the damping ratio.

Step response for different values of ζ:



Poles of an underdamped system ($0 < \zeta < 1$):



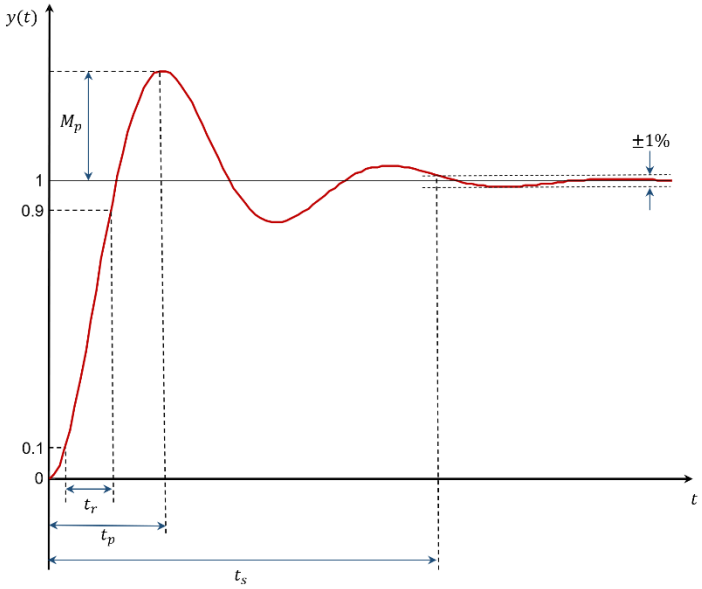
$$\alpha = -\zeta\omega_n$$

$$\beta = \omega_n\sqrt{1-\zeta^2}$$

Damped natural frequency:

$$\omega_d = \omega_n\sqrt{1-\zeta^2}$$

Step response of an underdamped system ($0 < \zeta < 1$):



- Rise time: $t_r \cong \frac{1.8}{\omega_n}$
- Settling time: $t_s = \frac{4.6}{\zeta\omega_n}$
- Peak time: $t_p = \frac{\pi}{\omega_d}$
- Overshoot: $M_p = e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}}$

We typically express the desired performance of a closed-loop system in terms of the rise time, settling time, etc., that is, in terms of the desired transient characteristics as well as the desired behavior in steady state (e.g., zero steady-state error when tracking step references).

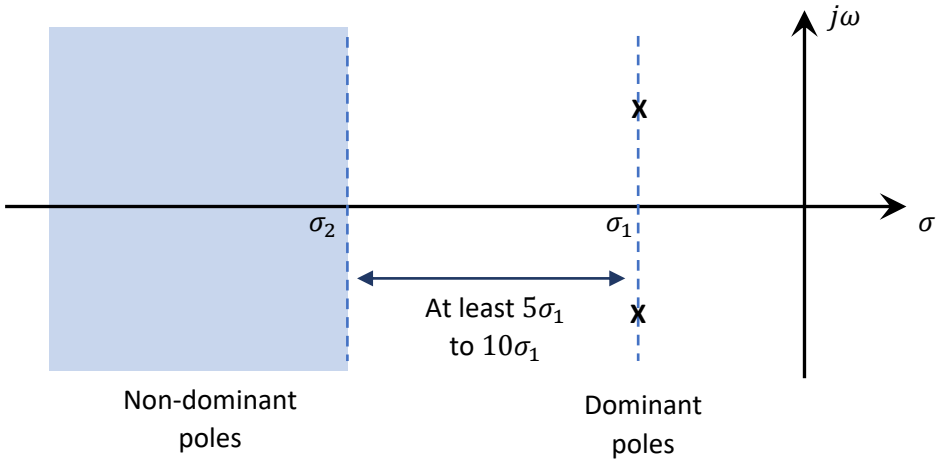
By exploiting the connection between the transient characteristics and the location of the poles of a second order system in standard form, we can set the dominant poles (the poles that would dominate the closed-loop behavior).

Let's work with a damping ratio $\zeta = 0.7$ and a settling time of $t_s = 1$ second. With those values, we can determine the dominant poles as follows:

$$t_s = \frac{4.6}{\zeta \omega_n} \quad \Rightarrow \quad \omega_n = \frac{4.6}{\zeta t_s} = \frac{4.6}{0.7 \times 1} = 6.5714 \ \text{rad/s}$$

Dominant poles: $-\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2} \ = \ -4.6 \pm j4.6929$

Since the order of the system to be controlled is $n = 4$, we need to determine 4 poles! Remember that a single pole at position $-a$ results in a term e^{-at} in the output. So, after some time, the poles with the largest real part will dominate the behavior. In order to choose the remaining 2 poles (nondominant poles) we can stick to the following heuristic rule: place them between 5 to 10 times farther away from the imaginary axis than dominant poles!



Nondominant poles: -46 and -50

Now, let's compute the controller gain $\boldsymbol{K} \in \mathbb{R}^{1\times 4}$ using the Ackermann's formula:

$$\boldsymbol{K} = [0 \quad 0 \quad 0 \quad 1]\mathcal{C}^{-1}\alpha_c(\boldsymbol{A}).$$

The "desired characteristic equation" $\alpha_c(s)$ is built with the desired closed-loop poles in the following way

$$\alpha_c(s) = (s + 4.6 + j4.6929)(s + 4.6 - j4.6929)(s + 46)(s + 50)$$

$$\alpha_c(s) = s^4 + 105.2s^3 + 3226.4s^2 + 25305.6s + 99322.4,$$

and this equation evaluated at $\boldsymbol{A}$ has the following form

$$\alpha_c(\boldsymbol{A}) = \boldsymbol{A}^4 + 105.2\boldsymbol{A}^3 + 3226.4\boldsymbol{A}^2 + 25305.6\boldsymbol{A} + 99322.4\boldsymbol{I}.$$

Finally, the gain of the controller is given by

$$\boldsymbol{K} = [-818.2441 \quad -513.8406 \quad -212.9450 \quad -72.0786].$$

Matlab code

```
% Determination of the desired closed-loop poles
dr = 0.7; %Damping ratio
ts = 1;   %Settling time
wn = 4.6/(dr*ts); %Natural frequency
alpha = -dr*wn;         %Real part of the dominant poles
beta = wn*sqrt(1-dr^2); %Imaginary part of the dominant poles
cl_poles=[alpha+beta*i alpha-beta*i  -46 -50]' %desired closed-loop poles

coeff_ec = poly (cl_poles); %Desired characteristic equation

% Computing K – First way
ec_A = polyvalm(coeff_ec,A); %Evaluation of the characteristic equation at A
K = [0 0 0 1]*inv(CO)*ec_A   %Ackermann's formula

% Computing K – Second way
K = acker(A,B,cl_poles)
```

Let's verify that the poles of the closed-loop system $\dot{\boldsymbol{x}} = (\boldsymbol{A} - \boldsymbol{BK})\boldsymbol{x}$ are placed in the desired locations. To this end we have to compute the eigenvalues of the matrix $\boldsymbol{A} - \boldsymbol{BK}$,

Matlab code

```
eig(A-B*K)
```

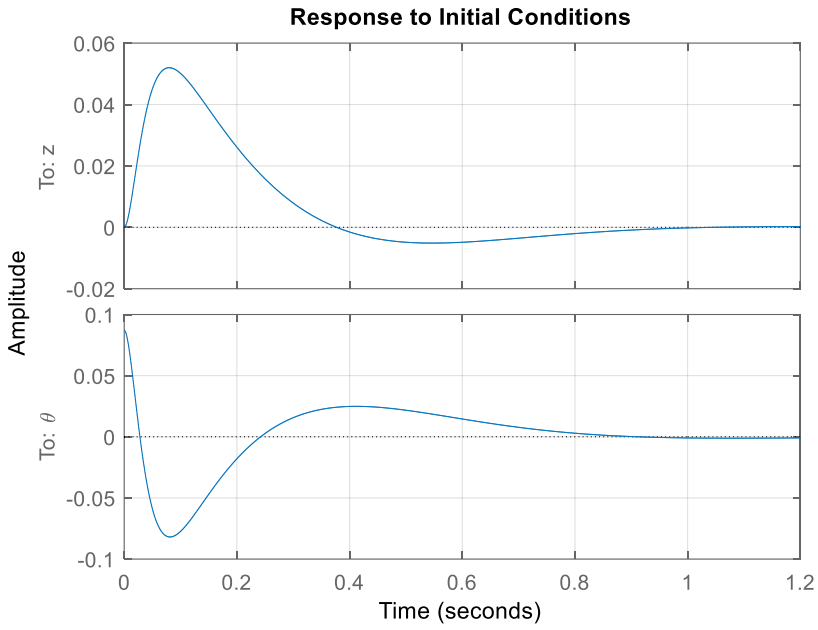
which are $-4.6 \pm j4.6929$, -46, and -50.

Now, it is time to carry out some simulations and evaluate the performance of the designed control system. In order to see the response of the closed-loop system to the initial condition $\boldsymbol{x}(0) = [z(0) \quad \theta(0) \quad \dot{z}(0) \quad \dot{\theta}(0)]^T = [0 \quad 5\pi/180 \quad 0 \quad 0]^T$ (initial rod angle of 5 degrees), we have to execute the following MATLAB code:

Matlab code

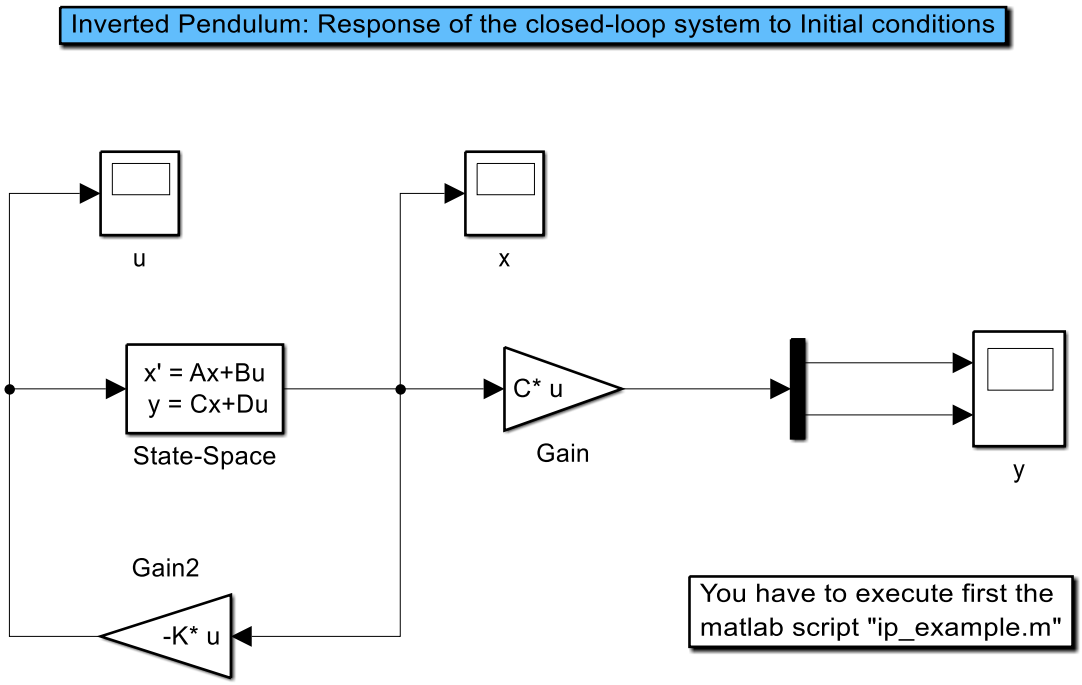
```
% Defining the closed-loop system
A_cl = A - B*K;
B_cl = [0;0;0;0];
C_cl = C;
D_cl = D;
sys_cl = ss(A_cl,B_cl,C_cl, D_cl)
sys_cl.OutputName={'z','\theta'};

%Setting the initial conditions
x0 = [0 5*pi/180 0 0]; %Initial rod angle = 5 degrees

%Plotting the response of the closed-loop system to the initial conditions
initial(sys_cl,x0)
grid
```

The response to initial conditions is given in the following figure.



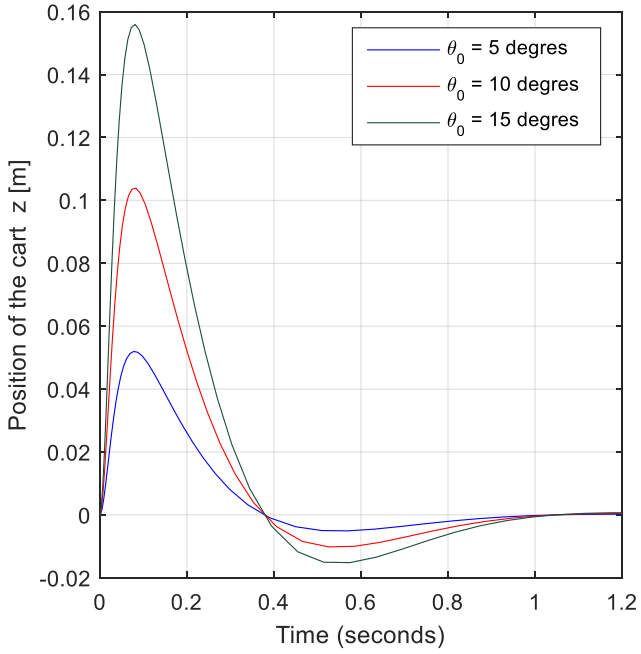
Notice that the closed-loop system reaches steady-state around 1 second as expected!

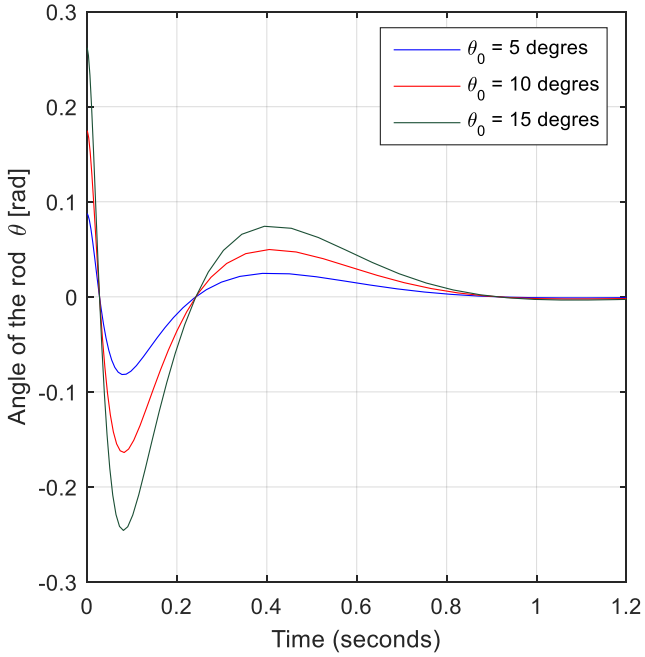
We can also use Simulink (which is a more flexible environment for performing simulations) to simulate the closed-loop system and evaluate its response to different initial conditions. Check the Simulink file "**ip_simulink.slx**"

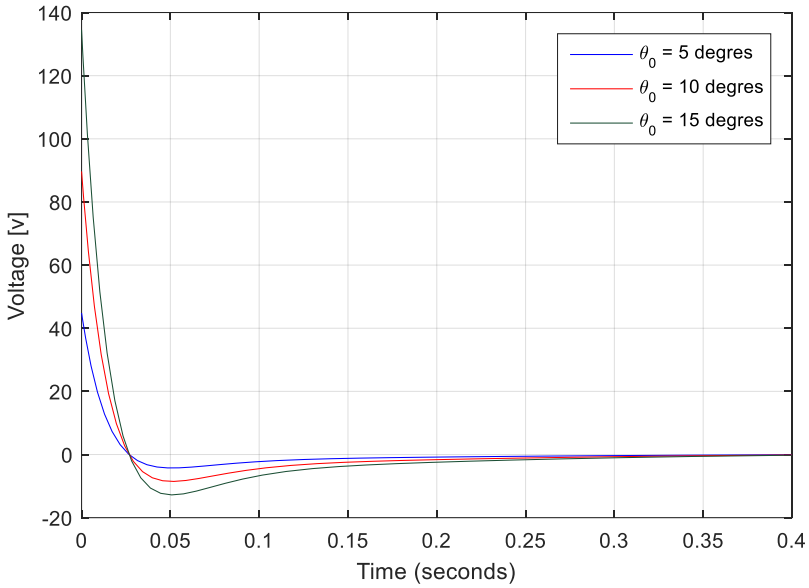


Recommendation: By default, the "gain block" in Simulink is set to perform element-wise matrix multiplication. So, make sure that the block is configured to perform matrix multiplication (Matrix K*u).

In the following figures, we case see the results of running the Simulink diagram with different initial rod angles ($\theta(0) = 5\pi/180$ rad, $\theta(0) = 10\pi/180$ rad and $\theta(0) = 15\pi/180$ rad).





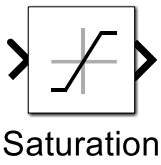


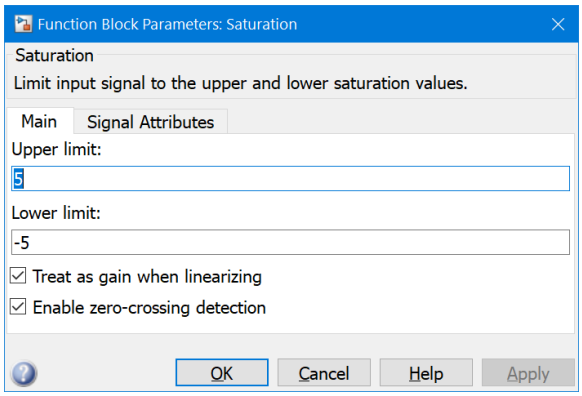
Notice that when the initial angle of the rod increases, the initial voltage (control action) applied to cart also increases, so that the cart can move fast enough to keep the rod upright!

Note: To set the initial angle of the rod, double click the “state-space” block and enter the initial state vector in the “initial conditions” textbox.

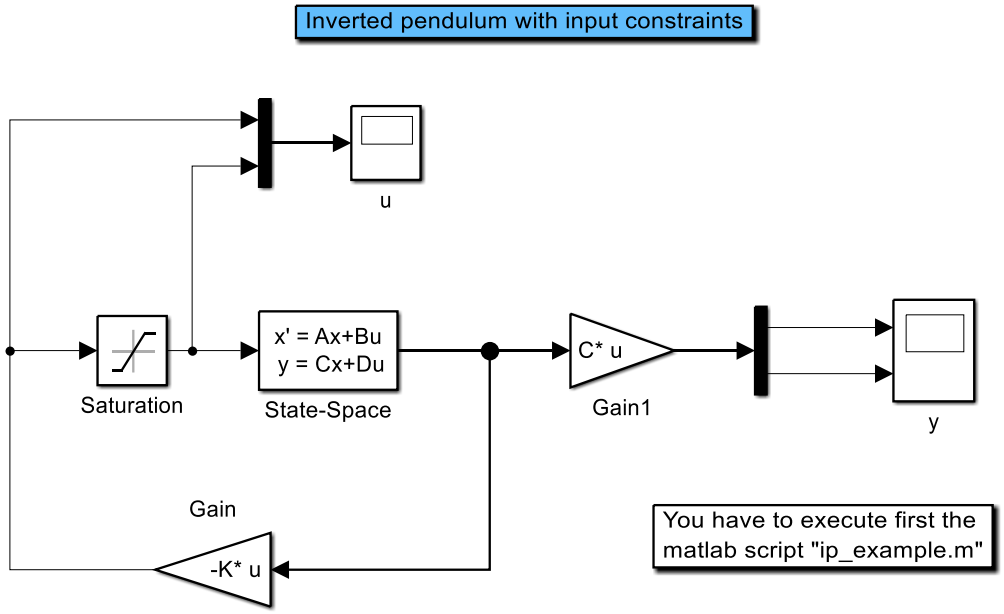
Considering the actuator constraints (Input constraints)

In order to make the simulations more realistic, let's consider the input constraints of the inverted pendulum (like the one we have in the Lab): the DC motor attached to cart can only work with voltages between -5V and 5V. Larger voltages are clipped. To model this constraint, we can use the block "Saturation" in Simulink.

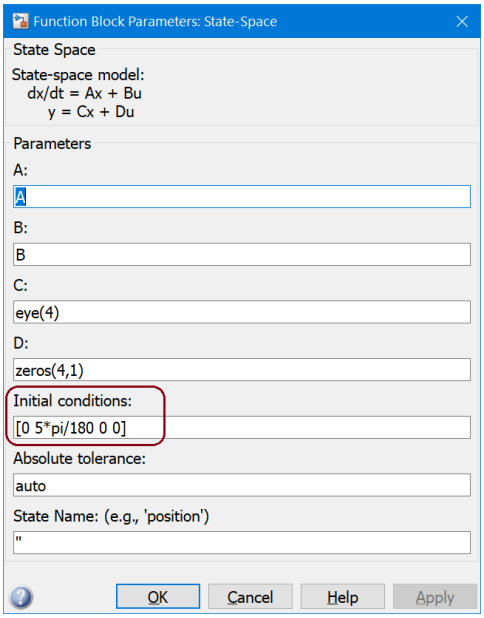


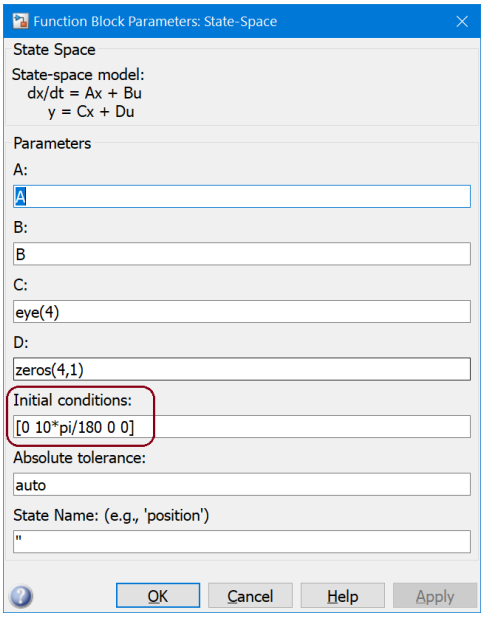


Simulink diagram that includes the input constraints (check the file: "**ip_simulink_saturator.slx**"):



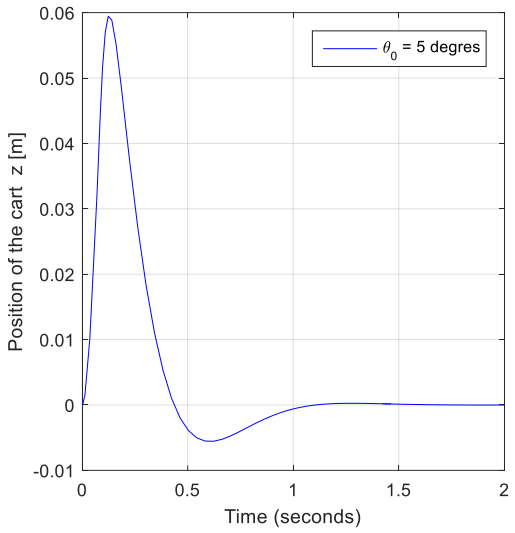
Let's carried out two simulations, one where the initial angle of the rod is $\theta(0) = 5\pi/180$, and the other one where the angle of the rod is $\theta(0) = 10\pi/180$.

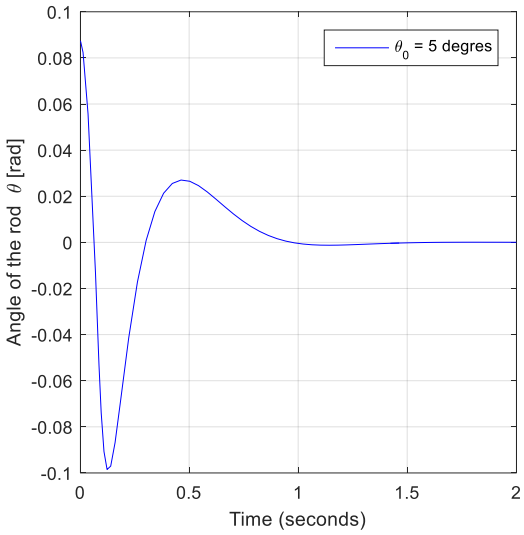


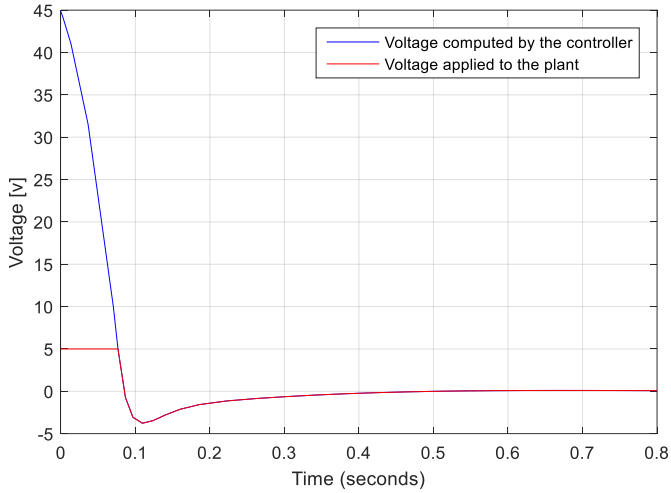


The results are as follows.

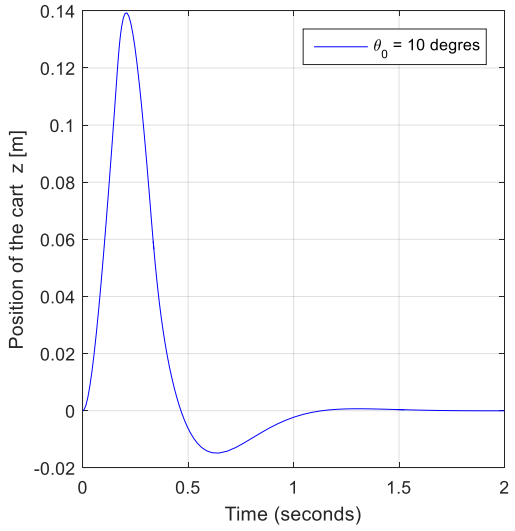
For $\theta(0) = 5\pi/180$:

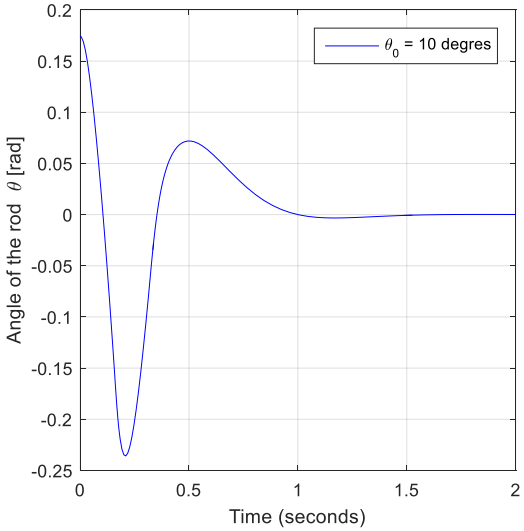


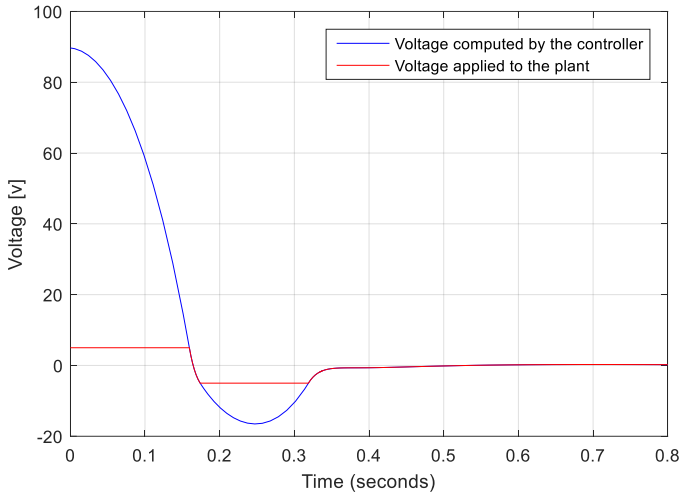




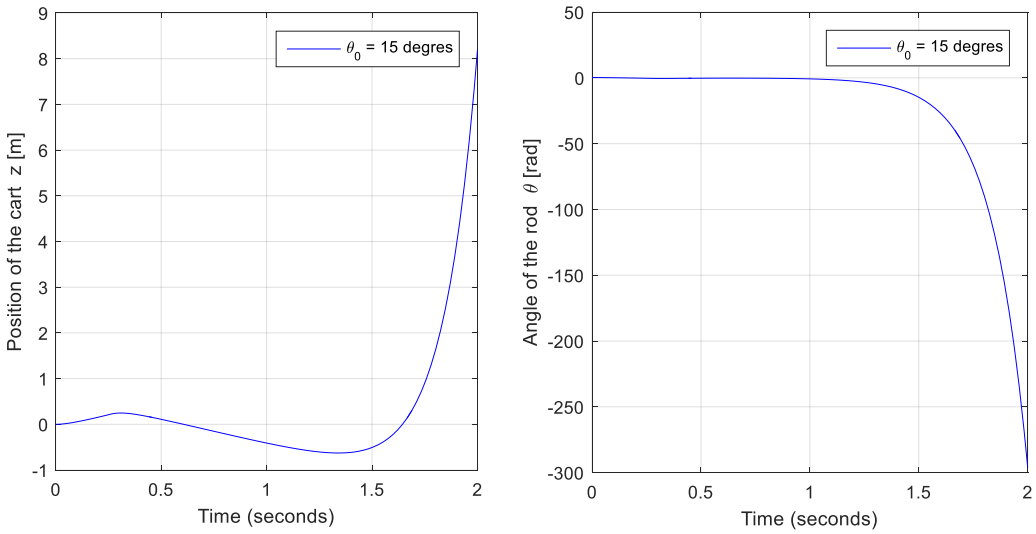
For $\theta(0) = 10\pi/180$:

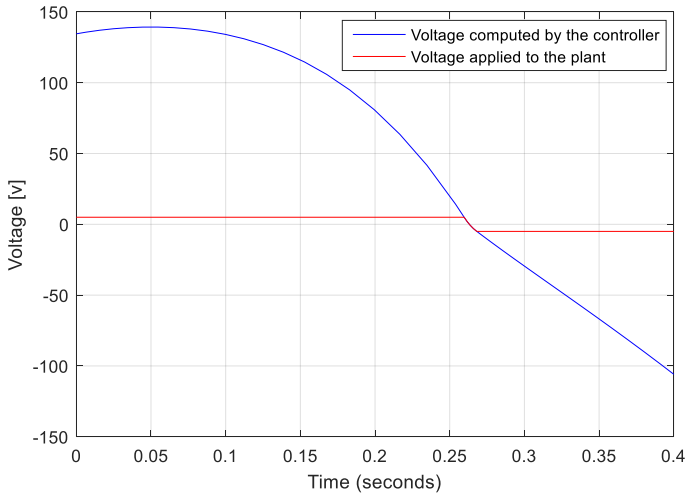






Notice that as the initial angle of the rod increases, the period in which the control actions are clipped also increases, which degrades the performance of the close-loop system (when compared with the unconstraint simulations). If we set the initial angle of the rod to $\theta(0) = 15\pi/180$, we get

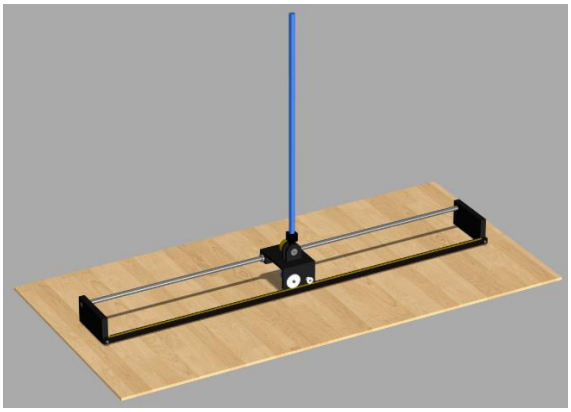




Since the actuator was saturated most of the time (is like having the plant in open loop), the closed-loop system became **unstable!** Keep in mind that the controller is not "aware" of the physical constraints of the plant.

Note: You can also reproduce the simulation results of this section using the Virtual inverted pendulum, which incorporates the physical constraints of the system, namely, voltage constraints and track constraints:

https://homes.esat.kuleuven.be/~maapc/VirtualControlLab/test11.html



- Set the numerical values of the controller gain $\boldsymbol{K} = [\mathrm{k}_1 \quad \mathrm{k}_2 \quad \mathrm{k}_3 \quad \mathrm{k}_4]$ to

$$\boldsymbol{K} = [-818.2441 \quad -513.8406 \quad -212.9450 \quad -72.0786]$$

- Set the "setpoint of the cart" to "Manual"
- In "initial conditions", set the rod angle (in degrees) to any value between -15 and 15.
- Click the "Start" button to start the simulation
- You can also perturb the system by pushing the rod. Just have fun!

What to do? Let's quickly design a less aggressive state feedback controller (a controller that generates smaller control actions/voltages)!

Again, let's work with a damping ratio $\zeta = 0.7$, but this time with a **settling time of $t_s = 2$ seconds**. With these specifications the dominant poles are:

$$t_s = \frac{4.6}{\zeta \omega_n} \quad \Rightarrow \quad \omega_n = \frac{4.6}{\zeta t_s} = \frac{4.6}{0.7 \times 2} = 3.2857 \ \text{rad/s}$$

Dominant poles: $-\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2} = -2.3 \pm j2.3465$.

Following the rule of thumb mentioned before, we place the nondominant poles at these locations:

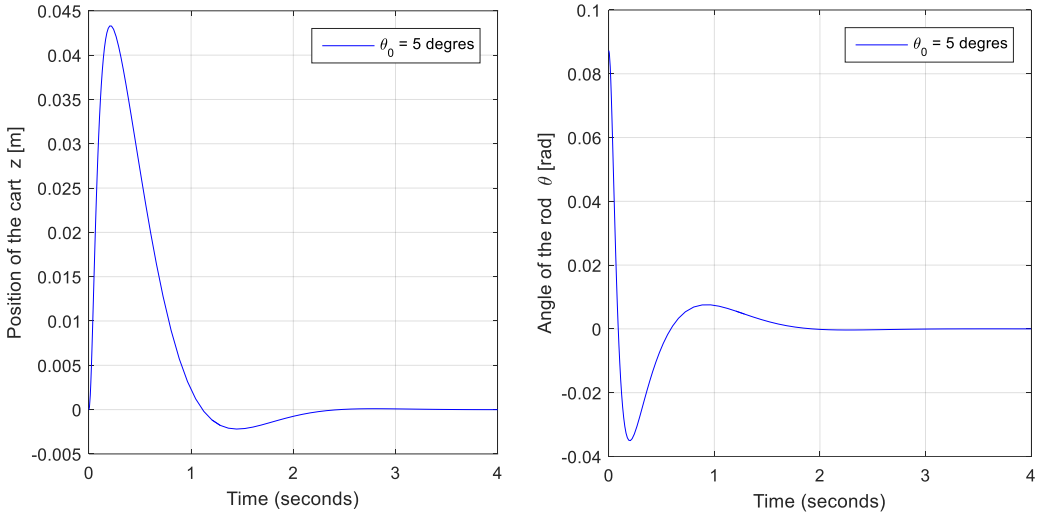
Non-dominant poles: -23 and -23.23.

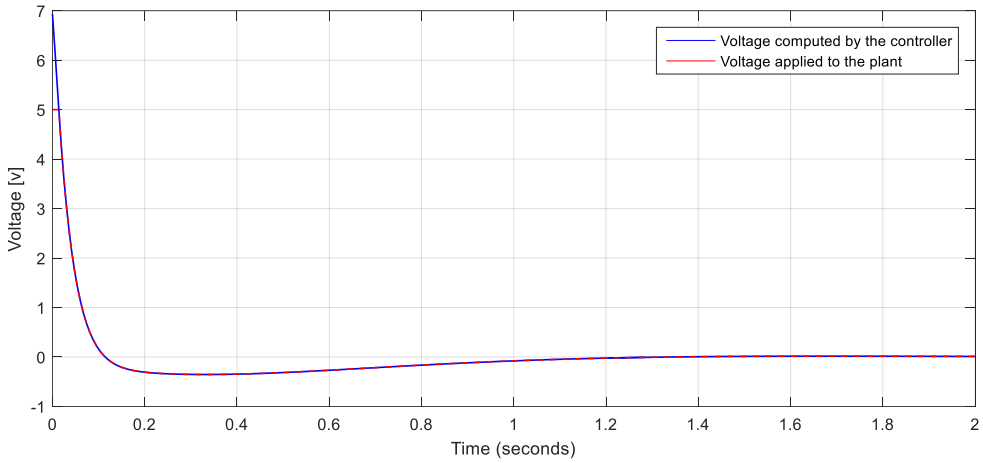
The new controller gain is given by

$$\boldsymbol{K} = [-47.5195 \quad -79.4617 \quad -28.8285 \quad -11.5334].$$

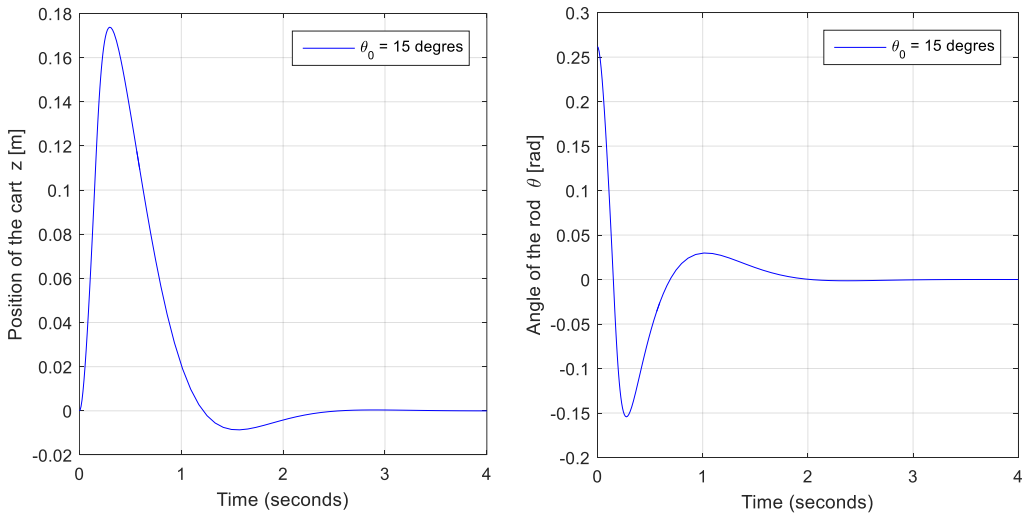
Observe that the entries of this matrix are smaller than the entries of the gain of the first design. In the following figures we can see the simulation results with this new controller.

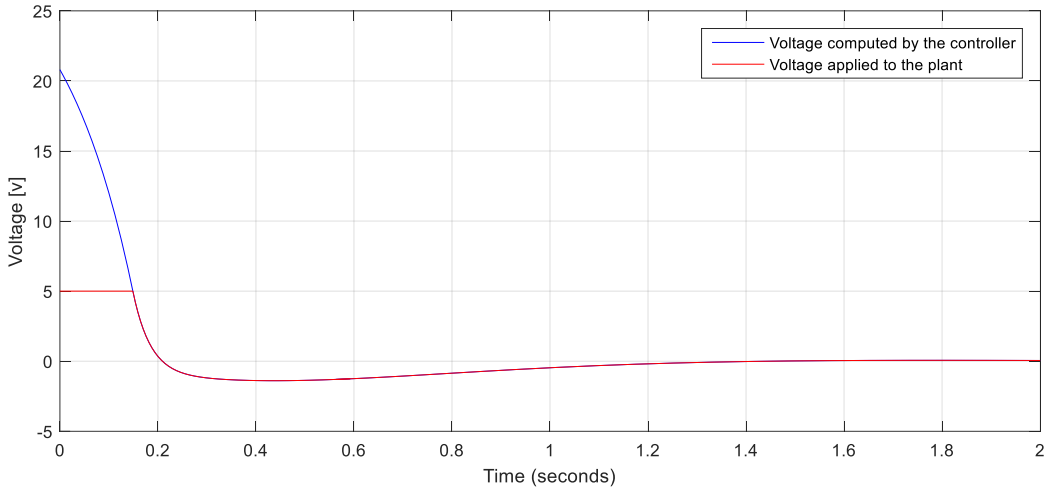
For $\theta(0) = 5\pi/180$:





For $\theta(0) = 15\pi/180$:





Clearly the control actions are way smaller and therefore the time the actuator is saturated is way smaller. We have designed a more robust controller. **At the end we always have to look for a good trade-off between performance and robustness!**

Do not forget to evaluate the new controller using the virtual inverted pendulum. It is more fun to accompany the plots of the signals with a 3D animation!

From this exercise it should be clear that we have to set realistic control objectives that consider the physical limitations of the plant. Takeaway message: when designing a control system, always keep an eye on the control actions! Unfortunately, this is something that people tend to forget.