Department ESR
Lecture: Fuzzy Control
Lab No. 1: Classical Controller Design

Objectives

This lab is concerned with the design and implementation of various controllers/compensators for the stabilization of an inverse pendulum. The compensators are designed using classical control engineering methodologies. Calculation and simulation is done in MATLAB.

Prerequisites

You are familiar with classical control engineering by root locus and pole placement design. You have basic experience with MATLAB and Simulink.

If you are short of MATLAB or Simulink basics you should consult one of the online MATLAB tutorials. Help can easily be obtained via the MATLAB Help Window: *MATLAB Help –> MATLAB –> Getting Started,* where you can browse through all essential MATLAB topics. For self study we also recommend the Control Tutorial for MATLAB from Carnegie Mellon University to be found at *http://www.engin.umich.edu/group/ctm/.*

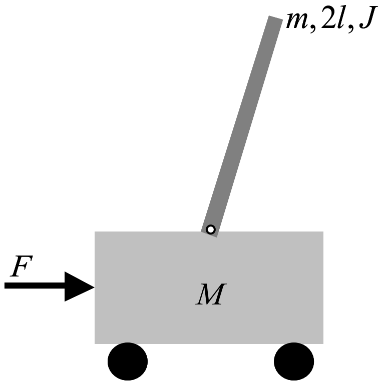
In case you need to refresh your knowledge on classical control theory, study a textbook such as Modern Control Systems, 9th edition, Richard C. Dorf and Robert H. Bishop, Prentice Hall or Modern Control System Theory and Design, 2nd edition, Stanley M. Shinners, John Wiley & Sons, In addition, we recommend the Control Tutorial for MATLAB, especially the PID and state space tutorials.

Model

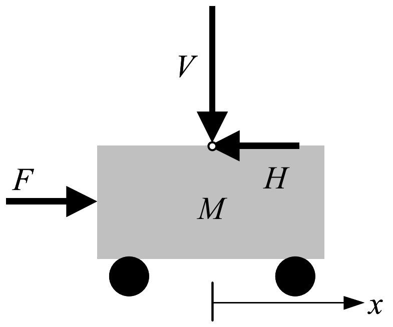
The figure below depicts a cart with an inverse pendulum (long thin pole) which can be controlled by the horizontal force F. In the following we derive the nonlinear equations of motion which afterwards are linearized around the vertical position. This operation point is defined by the angle $\theta = 0$. Friction is neglected throughout our analysis.

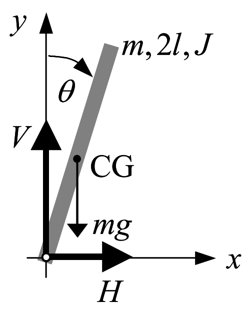
Notation of cart pole parameters

F	force applied to cart
M	mass of cart
m	mass of pole
l	half length of pole
J	inertia of pole



Analysis of the cart and the pole as free bodies gives:





For the cart, the equation of of forces in x-direction is

$$M\ddot{x} = F - H\ .$$

Denoting the coordinates of the joint and the pole's center of gravity by x, y and x_{cg}, y_{cg} respectively gives

$$\begin{aligned} x_{cg} &= x + l\sin\theta \\ y_{cg} &= y + l\cos\theta\ . \end{aligned}$$

The horizontal force H causes a translation of the pole in x-direction

$$m\ddot{x}_{cg} = H\ .$$

After differentiating x_{cg} twice the equation of motion of the cart with coordinate x results as

$$(M+m)\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = F\ .$$

The vertical and horizontal forces V and H cause a torque about the pole's center of gravity. V is calculated from the pole's force equation in vertical direction (using $\ddot{y}_{cg}$ and $\dot{y}_{cg} = 0$) according to

$$m\ddot{y}_{cg} = V - mg\ .$$

The torque equation of the pole about the center of gravity is (using clockwise orientation)

$$J\ddot{\theta} = Vl\sin\theta - Hl\cos\theta\ .$$

The equation of the pole's rotational motion is

$$(J + ml^2)\ddot{\theta} - mgl\sin\theta = -ml\ddot{x}\cos\theta\ .$$

Using the two equations of translation of the cart and rotation of the pole results in

$$\ddot{x} = \frac{1}{M+m}\left[F + ml\left(\dot{\theta}^2 \sin\theta - \ddot{\theta}\cos\theta\right)\right]$$

$$\ddot{\theta} = \frac{g\sin\theta - \cos\theta \dfrac{F + ml\dot{\theta}^2 \sin\theta}{M+m}}{l\left(\dfrac{4}{3} - \dfrac{m\cos^2\theta}{M+m}\right)}$$

applying $J = \frac{1}{3}ml^2$ as the inertia of a long thin pole.

Linearization about Operating Point

The nonlinear model of the inverse pendulum is already implemented in the Simulink model `cart_pole.mdl`. The force F is the input to the system, the four outputs are the four system states θ, $\frac{d\theta}{dt}$, x, $\frac{dx}{dt}$.

The physical parameters m, M and l can be specified as block parameters. The default values are $l = 1.0$, $m = 0.1$ und $M = 1.0$.

- Linearize the model `cart_pole.mdl` about the operating point $\theta = 0$ using MATLAB's `linmod` function, resulting in a LTI state space model. The linear model should be named `lti_cart_pole` for the sake of convention.
- Determine the step response of the linearized system. To which direction does the state θ change if the force F is positive? Which are the implications concerning the gain of a possible PD or PID controller?

PD/PID Controller Design

You are supposed to design a PD and a PID controller for stabilization of the pendulum. The design is supported by the MATLAB's SISO Design Tool `sisotool`. This tool encorporates graphical support of loop shaping by modification of Bode diagrams as well as root locus design. At first, the SIMO model `lti_cart_pole` is converted to a pure SISO model with θ as the single output:

```
ss_cart_pole=ss(lti_cart_pole);
plant = series(ss_cart_pole,ss([],[],[],[1 0 0 0]);
```

Next, the PD controller with its two parameters k and k_d is defined as an LTI transfer function:

```
kd = -1;
k  = -1;
pdcontroller = tf([kd k],[1]);
```

For the abovementioned parameters the closed loop is unstable, as it contains a pole in the right half plane.

Determine the controller parameters using `sisotool` such that the closed system becomes stable.

- Import the plant `plant` and the controller `pdcontroller` from workspace.
- Modify the controller gain k and the controller's zero (resulting from the D-part), thus stabilizing the closed loop. Analyze the disturbance performance of the closed loop: `Analysis -> Other Loop Responses...-> du to y`. Use a step disturbance. Is it possible to compensate this disturbance by means of a PD-controller? Determine the controller parameters k and k_d such that the output of the closed loop converges to a steady state after 2 seconds maximum (5% error region) with maximum 10 % overshoot.
- Which values k and k_d do you obtain in PD parameterization?
- Determine the static control error as the deviation of the pendulum angle from the vertical equilibrium for a step disturbance.

Now a PID controller is used, expanding the PD controller by an I-part.

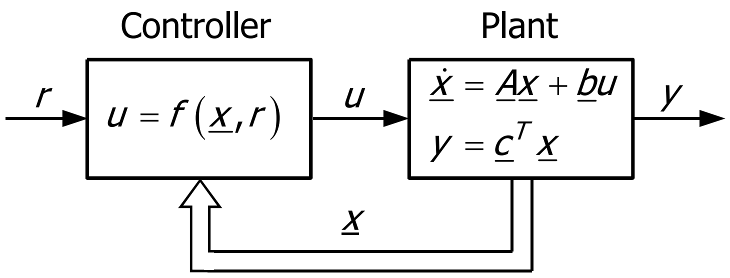
```
k  = ...;
kd = ...;
ki = ...;
pidcontroller = tf([kd k ki],[1 0]);
```

- Optimize the parameters k, k_d and k_i such that the closed loop is stable. If a step disturbance occurs the system output should reach a steady state after 5 seconds maximum. Is it possible to compensate the disturbance by means of a PID-controller? Determine the static control error. Which values of the PID parameters do you obtain?
- Implement the closed loop in Simulink by integrating the PID controller into the original exact nonlinear plant model `cart_pole.mdl`. Also add a limiter (Saturation Block) to the controller output F. Assume, that the operating range of the control variable *F* is confined to the interval [-5,5] Newton.
- For which initial angles θ_0 does the nonlinear pendulum remain stable when regulated by the PID controller originally designed in the linear domain?
- What is the behavior of the remaining states x and dx/dt?

State Space Controller / Pole Placement

Both the pole angle and the position of the cart can be stabilized by a linear state space controller. Again, we use the linearized model for controller design. In contrast to the output feedback of PID-control, in which only the output y is fed back, all inner states contribute to the static control law that determines the control variable F

$$F = -k_1\theta - k_2\dot{\theta} - k_3 x - k_4\dot{x} + r$$


Controller
Plant
r
$u = f(\underline{x}, r)$
u
$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{b}u$
$y = \underline{c}^T \underline{x}$
y
$\underline{x}$

- Determine the state space model and the eigenvalues of the linearized plant

```
[A,B,C,D]=linmod('cart_pole');
eig(A);
```

- Determine the controllability and observability matrices using MATLAB's `ctrb` and `obsv` functions. Verify whether the system is controllable and observable.

State feedback design allows you to arbitrarily place the poles of the closed loop. The only constraint in pole placement are the possible limitations of the control action u, which in turn limits the magnitude of the feedback gain vector $\mathbf{k}$. The feedback gain $\mathbf{k}$ is computed such that the eigenvalues of the closed loop's system matrix $(\mathbf{A}-\mathbf{bk})$ match some given λ_i.

- The eigenvalues of the closed loop are supposed to be located at $\lambda = [-2, -2, -1+i, -1-i]$. Determine the appropriate state feedback $F = -k_1\theta - k_2\dot{\theta} - k_3x - k_4\dot{x}$ using MATLAB's `acker` function.
- Integrate your state space controller with the nonlinear plant model in Simulink and analyze the disturbance performance of the closed nonlinear loop.
- Analyze the closed loop behavior for different initial pendulum configurations. The Simulink menu `Simulation->Simulation Parameter->Workspace I/O->Inital State` allows you to specify the initial system state.
- For which range of pendulum inclinations is the state space controller able to stabilize the pendulum and the cart? At which inclination does the system become unstable?