Controller Design: State Feedback Control
Cart-Pendulum System

Lab 05 - EGH445 Modern Control - 2020

Electrical Engineering & Robotics (EER)
Queensland University of Technology

July 13, 2020

Objectives

The objective of this Lab is to design state-feedback controllers for the cart-pendulum system and simulate the closed loop using Matlab-Simulink.

---

1 Inverted pendulum on a cart

We use the cart-pendulum system as a benchmark to design controllers based on state feedback. Figure 1 shows the idealised model of the system that consists of a pendulum of mass m and length ℓ attached to a cart of mass M_c. The pendulum moves under the action of the gravity (g) and the cart moves on the horizontal direction and is actuated by the control force F. The state-space model can be written as follows

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \\ \frac{x_4^2 \ell m \sin(x_2) - gm \sin(x_2)\cos(x_2) + F}{M_c + m - m\cos^2(x_2)} \\ \frac{-\ell m \sin(x_2)\cos(x_2) x_4^2 + g(M_c+m)\sin(x_2) - \cos(x_2) F}{\ell[M_c + m - m\cos^2(x_2)]} \end{bmatrix}, \tag{1}$$

where the states are

- x_1: the position of the cart,
- x_2: the angle of the pendulum,
- x_3: the velocity of the cart,
- x_4: the angular velocity of the pendulum.

We consider the values of the model parameters given in Table 1.

Table 1: Model parameters.

Parameter	value
m	0.15kg
M_c	0.4kg
ℓ	0.2m
g	9.81m/s^2

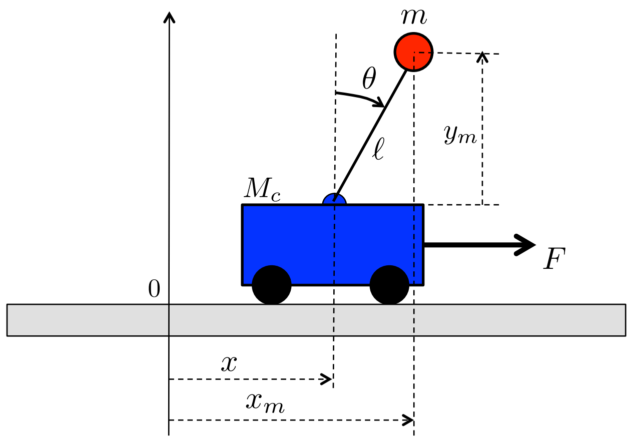

Figure 1: Cart-pendulum system.

1.1 Problem formulation.

In this lab, we consider the stabilisation problem of two desired equilibriums of the cart-pendulum system. The equilibriums are

$$\bar{x}_a = \begin{bmatrix} \bar{x}_{1a} \\ \bar{x}_{2a} \\ \bar{x}_{3a} \\ \bar{x}_{4a} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} ; \qquad \bar{x}_b = \begin{bmatrix} \bar{x}_{1b} \\ \bar{x}_{2b} \\ \bar{x}_{3b} \\ \bar{x}_{4b} \end{bmatrix} = \begin{bmatrix} 0 \\ \pi \\ 0 \\ 0 \end{bmatrix} . \tag{2}$$

The control objective is to find state-feedback controllers in the form

$$u = -K\, x,$$

to stabilise the equilibriums $\bar{x}_a$ and $\bar{x}_b$, where the input u is the control force F.

1.2 Control of the cart-pendulum at the upright position ($\bar{x}_a$).

The task in this section is to design a controller to stabilise the equilibrium $\bar{x}_a$. To do that, consider the linearised model of the cart-pendulum about $\bar{x}_a$:

$$\dot{\tilde{x}}_a = A_a \tilde{x}_a + B_a F, \tag{3}$$
$$y = C_a \tilde{x}_a + D_a F, \tag{4}$$

where

$$A_a = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{m\,g}{M_c} & 0 & 0 \\ 0 & \frac{g(M_c+m)}{\ell M_c} & 0 & 0 \end{bmatrix} ; \quad B_a = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{M_c} \\ -\frac{1}{\ell M_c} \end{bmatrix} ; \quad C_a = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} ; \quad D_a = 0_{4\times 1}. \tag{5}$$

Write a script that performs the following tasks:

a) Define the parameters of the model.

b) Compute the matrices of the linearised model.

c) Determine if the linearised model is controllable by computing the controllability matrix and checking its rank. Hint: use the command `ctrb`.

d) Compute the controller $u = -K_a\, \tilde{x}_a$ such that the eigenvalues of the (linearised) closed loop are $\lambda_1 = -3$, $\lambda_2 = -4$, $\lambda_3 = -5$ and $\lambda_4 = -6$. Hint: use the command `place`.

e) Simulate the linearised and nonlinear closed-loop systems:

   i) Create a Simulink model of the linearised system in closed loop with the state-feedback controller. Save your model as `CP_SFC_Lin_a_yourstudentnumber.slx`. An example of the Simulink model is shown in Figure 2.

ii) Create a Simulink model of the nonlinear system in closed loop with the state-feedback controller and select the control gain `K_SF=K_a`. Save your model as `CP_SFC_NLin_yourstudentnumber.slx`. An example of the Simulink model is shown in Figure 3.

iii) Export the states, the control input and the simulation time from Simulink to Matlab.

iv) Simulate both the linearised and the nonlinear closed-loop models with the cart stating at 0.2m and the pendulum at 20deg. Set the initial conditions of the velocities to zero. That is $x(0) = \begin{bmatrix} 0.2 & 20\pi/180 & 0 & 0 \end{bmatrix}^\top$. Suggestion for the simulation: use the fixed-step solver `ode4` and select 0.02 as step time.

f) Plot the results of the simulations in a figure that shows the time histories of the position of the cart, the angle of the pendulum, the velocity of the cart, the angular velocity of the pendulum and the control forces for both the linearised and nonlinear control systems. An example of the simulation results is shown in Figure 4.

g) Save the script as `CP_SFC_Lin_a_MainFile_yourstudentnumber.m`.

h) Run the script using different initial conditions for the simulation and analyse the results. The initial conditions should be defined in the script.

i) (Optional) Use the function `Cart_Pendulum_Animation.m` to create an animation of your control system.

Important: The plots and animation should be produced automatically when the script is executed without further user intervention.

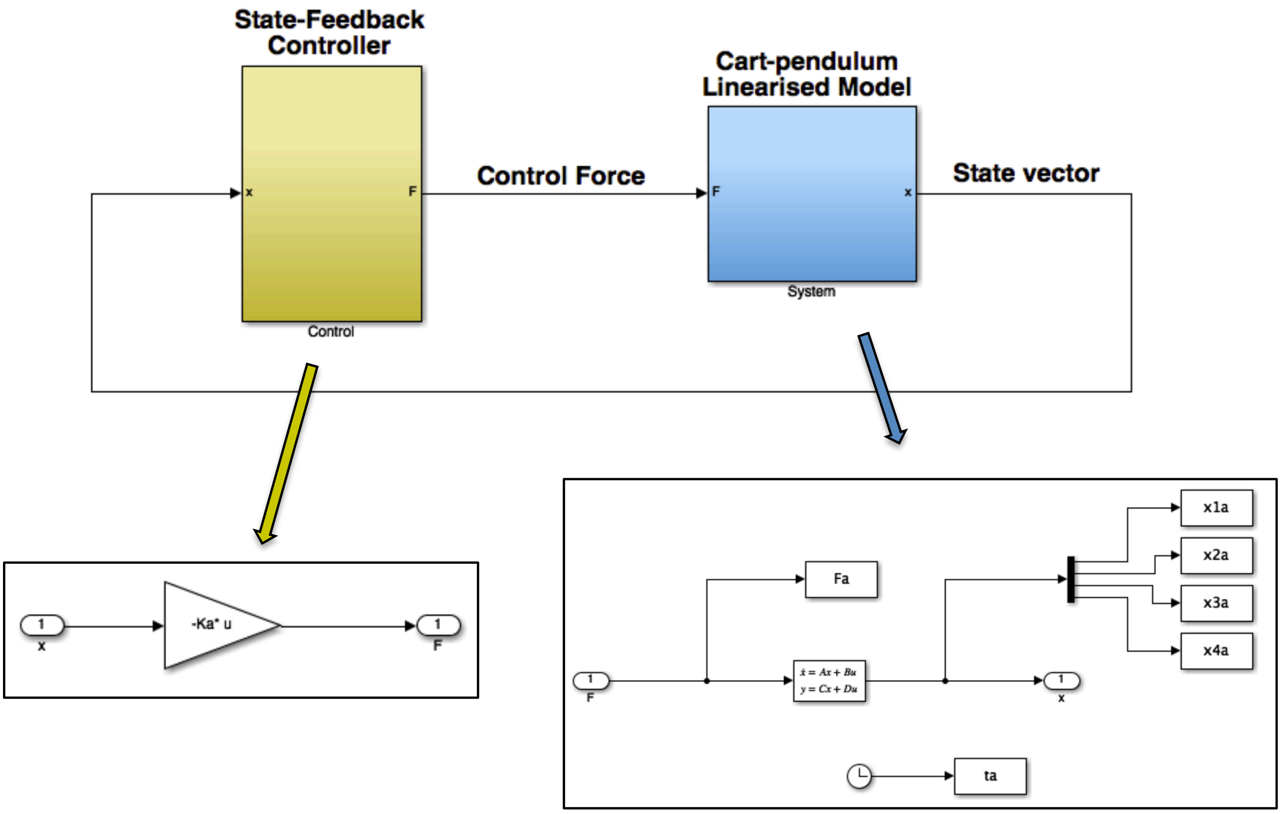

Figure 2: Linearised control system.

1.3 Control of the cart-pendulum at the down position ($\bar{x}_b$).

The task in this section is to design a controller to stabilise the equilibrium $\bar{x}_b$. To do that, consider the linearised model of the cart-pendulum about $\bar{x}_b$:

$$\dot{\tilde{x}}_b = A_b\tilde{x}_b + B_bF, \tag{6}$$

$$y = C_b\tilde{x}_b + D_bF, \tag{7}$$

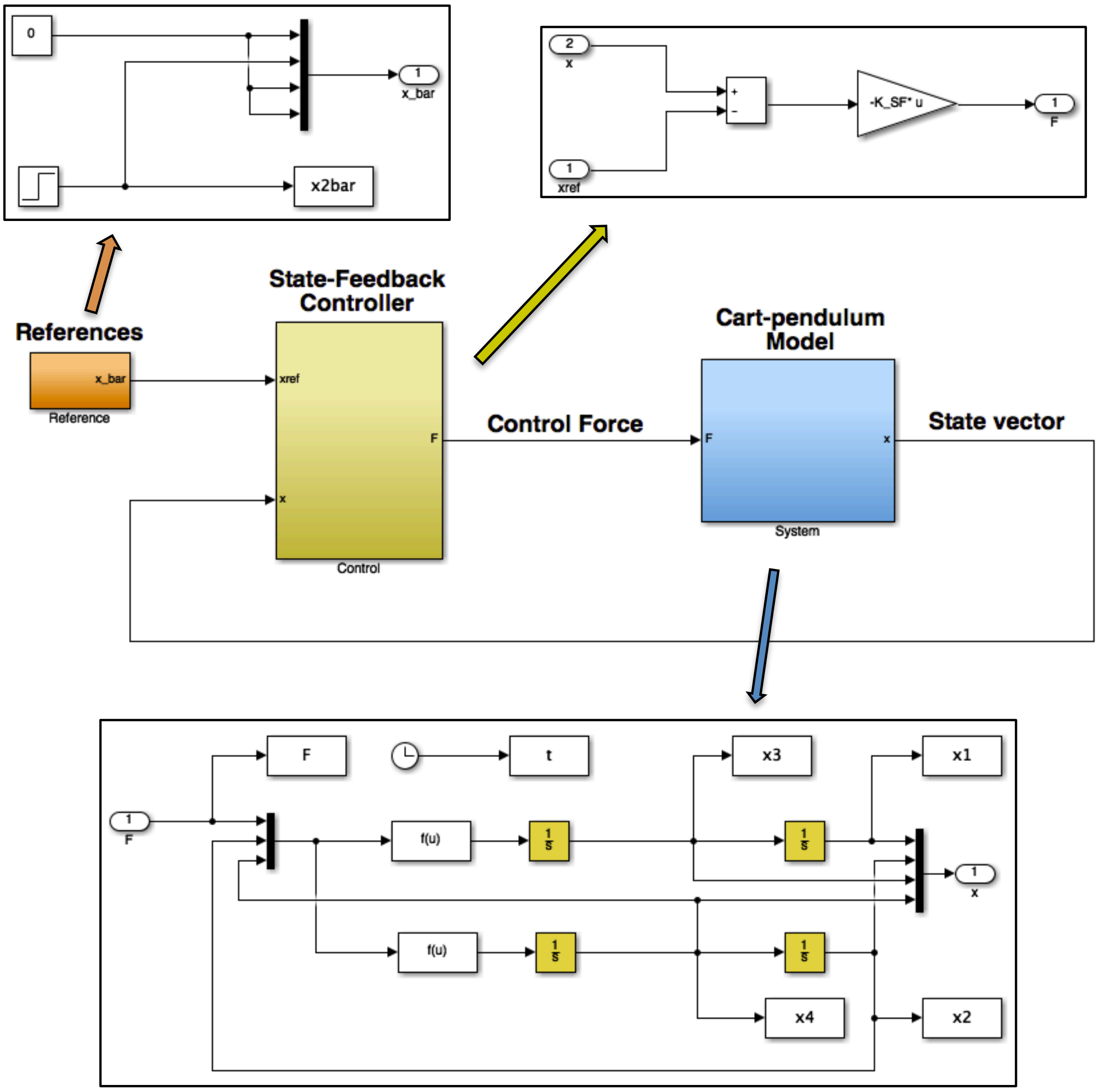

Figure 3: Nonlinear control system.

where

$$A_b = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{m\,g}{M_c} & 0 & 0 \\ 0 & -\frac{g(M_c+m)}{\ell M_c} & 0 & 0 \end{bmatrix}; \quad B_b = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{M_c} \\ \frac{1}{\ell M_c} \end{bmatrix}; \quad C_b = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad D_b = 0_{4\times 1}. \tag{8}$$

Write a script that performs the following tasks:

a) Define the parameters of the model.

b) Compute the matrices of the linearised model.

c) Determine if the linearised model is controllable by computing the controllability matrix and checking its rank.

d) Compute the controller $u = -K_b\,\tilde{x}_b$ such that the eigenvalues of the (linearised) closed loop are $\lambda_1 = -3$, $\lambda_2 = -4$, $\lambda_3 = -5$ and $\lambda_4 = -6$.

e) Simulate the linearised and nonlinear closed-loop systems:

   i) Create a Simulink model of the linearised system in closed loop with the state-feedback controller. Save your model as `CP_SFC_Lin_b_yourstudentnumber.slx`.

   ii) Reuse the Simulink model of the nonlinear system in closed loop with the state-feedback controller you created in Section 1.2 (`CP_SFC_NLin_yourstudentnumber.slx`) and select the control gain `K_SF=K_b`.

   iii) Export the states, the control input and the simulation time from Simulink to Matlab.

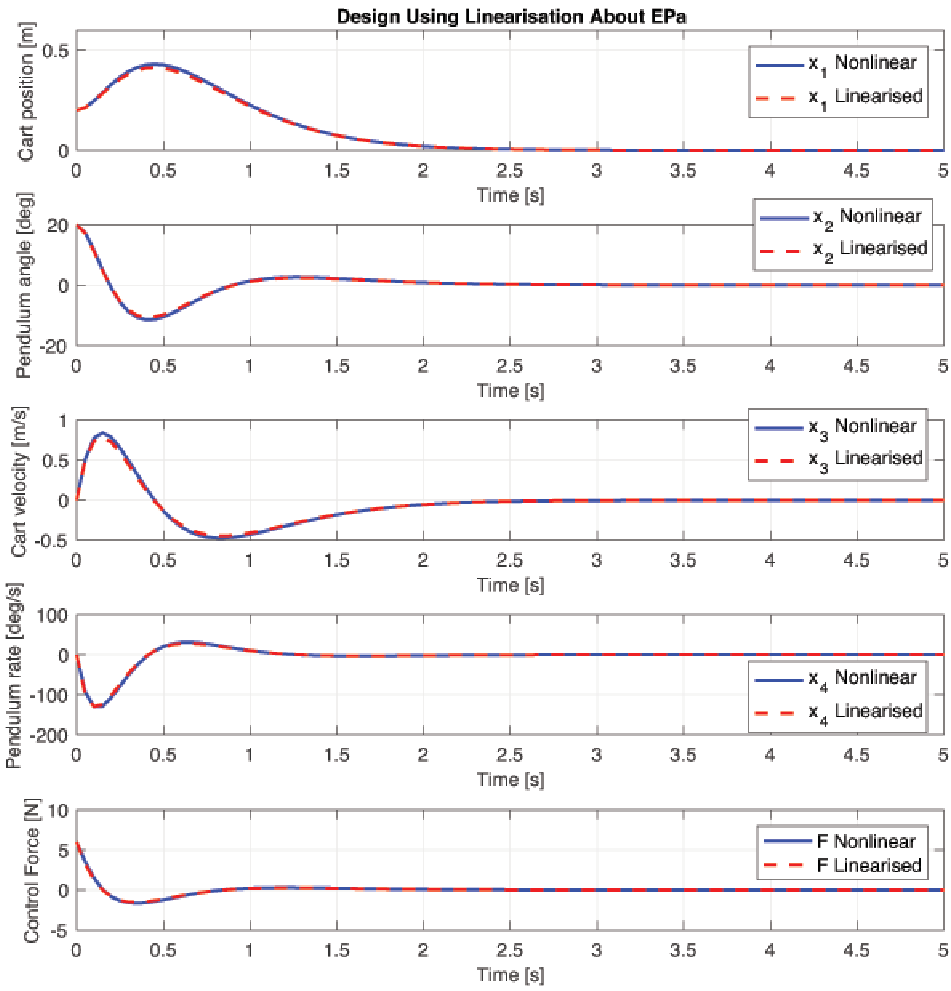

Figure 4: Time histories of the states and input.

iv) Simulate both the linearised and the nonlinear closed-loop models with the cart stating at 0.4m and the pendulum at 200deg. Set the initial conditions of the velocities to zero. That is $x(0) = \begin{bmatrix} 0.4 & 200\pi/180 & 0 & 0 \end{bmatrix}^\top$ (adapt the initial conditions for the linearised model). Suggestion for the simulation: use the fixed-step solver `ode4` and select 0.02 as step time.

f) Plot the results of the simulations in a figure that shows the time histories of the position of the cart, the angle of the pendulum, the velocity of the cart, the angular velocity of the pendulum and the control forces for both the linearised and nonlinear control systems. An example of the simulation results is shown in Figure 5.

g) Save the script as `CP_SFC_Lin_b_MainFile_yourstudentnumber.m`.

h) Run the script using different initial conditions for the simulation and analyse the results. The initial conditions should be defined in the script.

i) (Optional) Use the function `Cart_Pendulum_Animation.m` to create an animation of your control system.

Important: The plots and animation should be produced automatically when the script is executed without further user intervention.

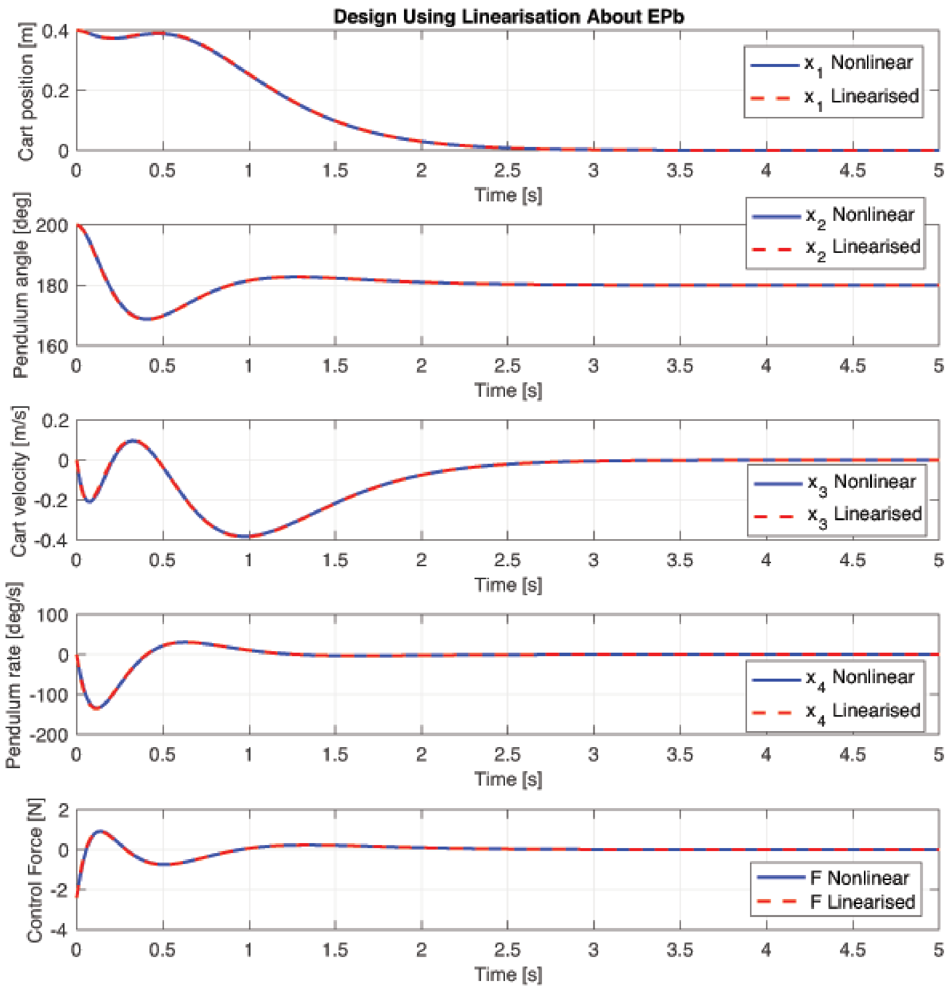

Figure 5: Time histories of the states and input.