Practical Course Matlab/Simulink

Control System Toolbox

Table of Contents

Table of Figures

0 General Information and Advice

The following exercises cover MATLAB's Control System Toolbox. They increase in difficulty and they also build upon each other, so you should start with the first exercise, continue with the second and so on. Basic continuous-time models, interconnections and functions are presented in the first exercise. The second exercise extends the scope to discrete-time models and model transformations. Tunable models and linear analysis are covered by the third exercise. The fourth exercise introduces control design functionalities and the final exercise applies most of the Control System Toolbox functions to the line-tracking robot considered in this lab course.

It is recommended that you write the MATLAB code you produce during this session into an M-File (MATLAB script). This makes it easier for your supervisor to help you in case of problems. You can also save and keep the file for your records.

At the beginning of each exercise, delete all existing variables.

1 Linear Dynamic Models and Interconnections

This first exercise introduces linear dynamic models and basic functions of the toolbox, like model interconnection. The example control loop is illustrated in the following block diagram (Figure 1-1).

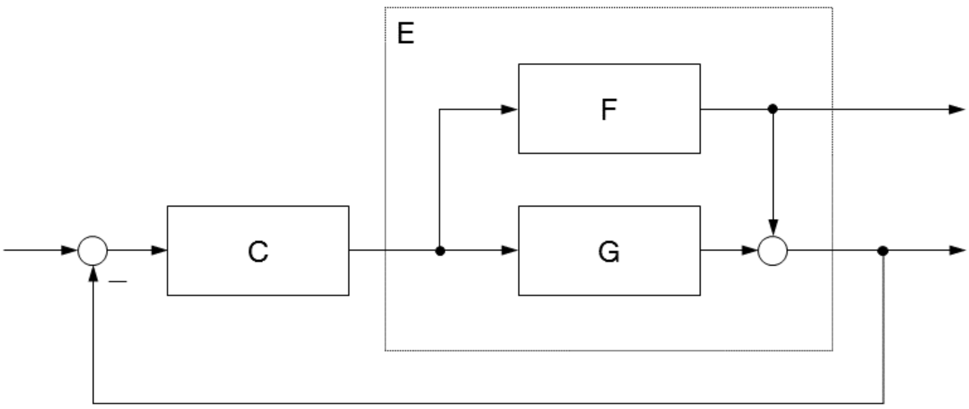


Figure 1-1: Block diagram of the example control loop

Exercise (8 points, 1 point each)

(1) Create a zero-pole-gain model **F** with a zero at 18, two poles $-3 \pm 2i$ and a gain of $4/2225$.

(2) Create a transfer function model **G** according to the following function

$$G(s) = \frac{(23 - s) \cdot 4/2225}{(s^2 + s + 4.25)} e^{-0.05s}$$

(3) Transform the model **G** to zero-pole-gain form **G_zpk**. What is the numerator of **G_zpk**?

ANSWER:	

(4) Connect the models **F** and **G** to assemble the model **E**. (Hint: use **parallel** and use matrix notation.)

(5) Create a PID controller **C** (parallel form) with $K_p = 3.16$, $K_i = 15.9$ and $K_d = 0.156$

(6) Make a series and a feedback connection of **`C`** and **`E`** to assemble the model **`closed_loop`**. Note: the resulting model should have one input and two outputs.

(7) Determine the order of the entire system **`closed_loop`**.

ANSWER:	

(8) Is the transfer from input 1 to output 2 of the mode **`closed_loop`** proper?

ANSWER:	

2 Discrete-Time Models and Advanced Model Transformation

This exercise extends the concepts presented in the first exercise. You create and handle discrete-time models, you perform model order reduction and frequency separation. The step response plot as a linear analysis tool in the time-domain is introduced as well.

Exercise (10 points, 1 point each)

(1) Create a 5th-order weighted moving average filter, given by the transfer function

$$H(z) = \frac{1}{15}(1 \cdot z^{-4} + 2 \cdot z^{-3} + 3 \cdot z^{-2} + 4 \cdot z^{-1} + 5 \cdot z^{0})$$

Let the sampling rate be $50\ Hz$. First, use the function **`filt`**, then use the function **`tf`**.

(2) Create a new system **`H_us`**, which is the filter H upsampled by a factor of 2.

(3) Create a new system **`H_rs`**, which is the filter H resampled at a sampling rate of $100\ Hz$. The model order is to be conserved! (Hint: to avoid the error message, change the resampling method by handing over **`'tustin'`** as a third function argument.)

(4) Create three separate figures with the step response of each system **`H`**, **`H_us`** and **`H_rs`**. Which two systems have the same step response?

ANSWER:	

(5) Create a model **`Y`** with poles at -10, -3, -1 and -1.

(6) Find a balanced realization **`Yb`** of the system **`Y`**.

(7) Use the information from the balanced realization to reduce the model order. Eliminate the two least important modes to obtain the reduced model **`Yr`**.

(8) Create one single figure with the step responses of **`Y`** and **`Yr`**. Compare.

(9) Make a frequency separation at $2\ rad/s$ to obtain slow and fast dynamics **`Ys`** and **`Yf`**.

(10) Create one single figure with the step responses of **`Ys`**, **`Yf`** and **`Ys+Yf`**. Note that the sum of slow and fast dynamics is the original system **`Y`** again.

3 Tunable Models and Linear Analysis

This exercise shows you how to analyze the linear dynamic characteristics of a tunable model.

Exercise (9 points, 1 point each)

(1) Create two tunable parameters **`w`** and **`z`** with initial values 3 and 0.9.

(2) Create a generalized state-space model `Q` with

$A = \begin{bmatrix} -2wz & -w^2 \\ 1 & 0 \end{bmatrix}$, $B = \begin{bmatrix} w^2 \\ 0 \end{bmatrix}$, $C = [0 \quad 1]$ and $D = 0$

(3) Create three different samples of `Q`, by varying the parameter **`z`** as follows: $(0.1, 0.7, 1)$

The resulting matrix of models shall be called **`Qsample`**.

(4) Plot the Bode diagram of all three sample models in one single figure. (Hint: only one single function call with one argument is required.)

(5) Visualize the poles of `Q` with **`z=1`** in the complex plane.

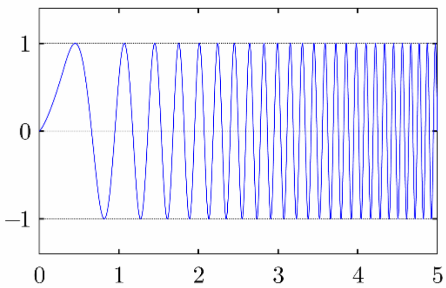
(6) What is the damping and the frequency of the poles of `Q` with **`z=0.7`**?

ANSWER:	

(7) What is the peak gain in decibels of `Q` with **`z=0.1`**?

ANSWER:	

(8) Create a chirp signal. A chirp signal is a sinusoidal wave with increasing frequency, as shown below.



You can do this by first specifying a time vector, containing 4000 linearly spaced values from 0 to 40. Then specify the input vector as follows:

$$\text{input} = \sin(\omega \cdot t) = \sin\big((t/10) \cdot t\big)$$

(9) Make a linear simulation of `Q` with **`z=0.1`** with the chirp signal as input. Observe how amplitude and phase of the simulated output signal change! Using the plot, visually determine the value of the simulated output at $t = 22.2s$!

ANSWER:	

4 Control Design

Now that you know how to create, transform and interconnect continuous and discrete, tunable and non-tunable models, this exercise teaches you how to perform control design using those models.

Exercise (10 points, 1 point each)

(1) Create a linear model $G(s) = \frac{1}{0.25s^2+s+1}$

(2) Use automatic PID tuning with default settings to find a proportional controller that raises the crossover frequency to $2.5\ rad/s$. What is the phase margin of the resulting control loop?

ANSWER:	

(3) Open the PID Tuner with the same model G. Tune a PID controller with the desired response time set to $0.1\ s$ and the transient behavior as aggressive as possible, without exceeding 10% overshoot. (Hint: use the “Show Parameters” window.) What are the resulting controller parameters?

ANSWER:	$K_P =$ $K_I =$ $K_D =$

(4) Open the SISO Design Tool with the model G.

(5) Choose the third control architecture, with a feedforward controller F that is parallel to the feedback controller C.

(6) In the feedback controller C, place a pole at -6 and a zero at -1

(7) Increase the gain until reaching a phase margin of 60°. (Hint: use graphical tuning.) What is the resulting transfer function of C?

ANSWER:	$C(s) = \frac{\quad}{\quad}$

(8) Let F be a simple gain with a value of 0.3.

(9) Create a step response analysis plot of the closed loop (r to y) and determine the time of the peak amplitude. (Hint: right click in the plot window -> characteristics -> peak response. Then hover over the blue dot to see peak response parameters.)

ANSWER:	

(10)Feel free to experiment with the tool.

5 Moving Robot – Wheel Speed Control

In this last exercise you will apply some of the Control System Toolbox functionalities to the problem of wheel speed control for the moving robot. Each of the two wheels of the robot is driven by an electric motor through a set of gears, as shown in Figure 5-1.

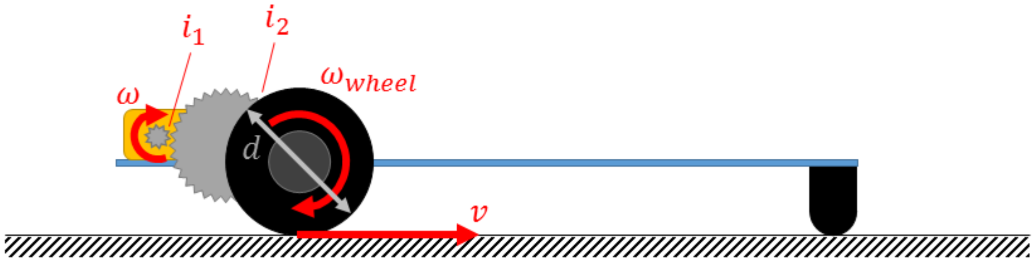


Figure 5-1: Side-view of robot with engine, gears and wheel

The relationship between motor command u and wheel velocity v can be written as a block diagram, like shown in Figure 5-2. Here, K_T is the motor torque gain, T is the motor's time constant, i_1 and i_2 are the gear transmissions and d is the wheel diameter.

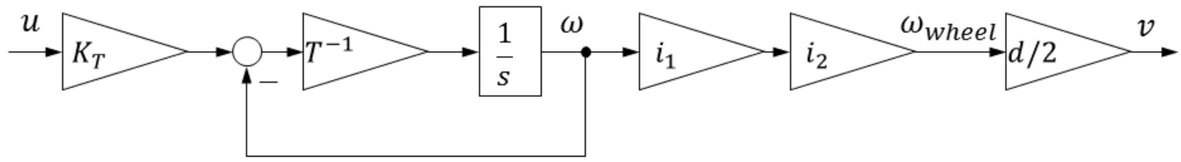


Figure 5-2: Block diagram of the wheel speed transfer function

Values for all these parameters that well represent the actual robot are given in the following list. Your task is to model this system in MATLAB and to find a controller that enables good reference tracking.

$$
\begin{aligned}
K_T &= 3.45 \\
T &= 0.001 \\
i_1 &= 1/6.5 \\
i_2 &= 1/5.6 \\
d &= 0.038
\end{aligned}
$$

Exercise (6 points, 3 point each)

(1) Create a continuous-time transfer function from u to v, called `G_wheel` in MATLAB.

(2) Design a PI controller for `G_wheel` (see Figure 5-3) with a bandwidth of 9 rad/s and a phase margin of 90°. Note down the resulting controller gains!

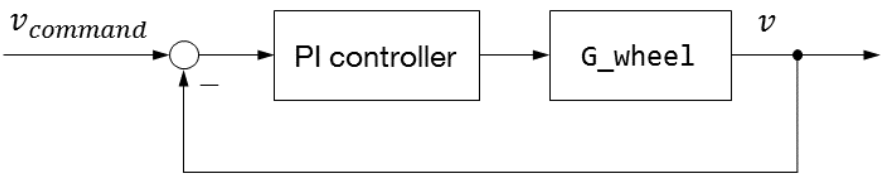


Figure 5-3: Block diagram of the closed wheel speed control loop

GAINS:	K_p:	K_i: