TA Solutions, Computerpracticum 2 Control Engineering

Position Control of a Coreless DC Motor

1 Introduction and objective

In the first computerpracticum you have investigated the dynamic behavior of a coreless DC motor with help of MATLAB and Simulink. The objective of this assignment is to control the position $\theta(t)$ of the inertial load (pulley) by regulating the input voltage $u(t)$ (see Figure 1).

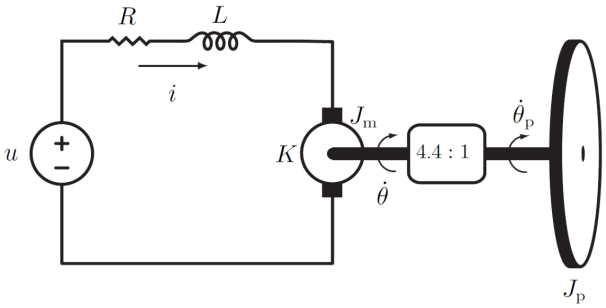

Figure 1: Schematic representation of a geared DC motor

- **The first four exercises concern the derivation of the transfer function and specifications for the system. It is required to prepare these exercises at home before the session!**
- **A start-up file including all parameters can be found on Nestor under `Assignments > Practicum`. You are kindly requested to use this file throughout the practicum, to allow the teaching assistants to quickly check your progress.**

2 System Model and Design Specifications

A first step in designing and evaluating a good controller is to derive a sufficiently accurate system model. In the previous assignment you have found two state space models. The first (second order) state space model has the angular velocity $\dot{\theta} = \dot{q}_m$ as an output and is given by

$$\dot{x} = \begin{pmatrix} 0 & \frac{K}{J_t} \\ -\frac{K}{L} & -\frac{R}{L} \end{pmatrix} x + \begin{pmatrix} 0 \\ \frac{1}{L} \end{pmatrix} u, \quad y = \begin{pmatrix} 1 & 0 \end{pmatrix} x, \tag{1}$$

with $x = (x_1, x_2) = (\dot{q}_m, \dot{q}_e)$. The second (third order) model has the angle $\theta = q_m$ as an output and is given by

$$\dot{x} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & \frac{K}{J_t} \\ 0 & -\frac{K}{L} & -\frac{R}{L} \end{pmatrix} x + \begin{pmatrix} 0 \\ 0 \\ \frac{1}{L} \end{pmatrix} u, \quad y = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} x, \tag{2}$$

with $x = (x_1, x_2, x_3) = (q_m, \dot{q}_m, \dot{q}_e)$. Recall that the motor parameters are given by $R = 2.32\Omega$, $L = 0.24 \cdot 10^{-3} H$, $K = 0.0234^{Nm}/_{A}$, $J_t = J_p/4.4^2 + J_m = 5.9301 \cdot 10^{-5}$. It is desired to have the pulley take positions between $\pm 2rad$. The system should have a zero steadystate error to a step change in the input voltage, i.e., $e_{ss} = 0$. It should have a rise time $t_r \leq 20ms$. Furthermore, it should have a settling time $t_s \leq 50ms$ and a percent overshoot $M_p \leq 4\%$.

Preparation exercises

Exercise 1. Transfer functions DC motor

Based on (1)-(2) determine

- The transfer function $H_{\dot{\theta}u}(s)$ from input u to output $\dot{\theta}$.
- The transfer function $H_{\theta u}(s)$ from input u to ouput θ, and express it in terms of $H_{\dot{\theta}u}(s)$.

Exercise 2. Reformulation of specifications (time domain)

The controller design in this practicum will result in a second order system of the form

$$H^{cl}(s) = \frac{\alpha\,(s+\beta)}{s^2+\gamma_1 s+\gamma_2}. \tag{3}$$

Sstem (3) may be approximated by a standard second order system of the form

$$\tilde{H}^{cl}(s) = \frac{\omega_0^2}{s^2+2\zeta\omega_0 s+\omega_0^2}. \tag{4}$$

However, neglecting the zero at $s=-\beta$ results in a larger overshoot than the approximated system (4) indicates. Using the approximation (4) allows us to use the design-rules for standard second order systems, which provide a good starting point for the controller design.

In computerpracticum 1 you designed a state feedback controller to assign the eigenvalues of the second order system 1 to $-100 \pm 100i$.

- Calculate the natural frequency ω_0 and damping ratio ζ corresponding to the eigenvalue locations $-100 \pm 100i$.
- Use Table 6.1 (page 185) in the book to estimate the rise time t_r, settling time t_s, and overshoot M_p.

Exercise 3. Model reduction

In many practical cases the armature inductance L of the DC motor is very small. This means that the electrical time constant can be neglected in the controller design process. Find the reduced 2^{nd} order transfer function $H_{\theta u}^{ro}(s)$ from $H_{\theta u}(s)$.

Exercise 4. Reduced model (state space form)

We will design an output feedback controller for the reduced model, but for that, we need to transform the model back into state space form.

Do so, by setting $L = 0$ in the equations of motion and deriving a new state space form for $x = (q_m, \dot{q}_m)$.

MATLAB

Please present your results in one of the following ways

1. **Showing the MATLAB command line output directly in a structered way**
2. **Copy-pasting the command line output into Microsoft Word**
3. **Writing down the command line output in the boxes below the questions in this manual**

Plots should be stored in separate figure-windows, such that the teaching assistant can check your results easily

Exercise 5. Model reduction (continued)

Use MATLAB to determine the Bode plots for both the full-order system $H_{\theta u}(s)$ and the reduced-order system $H^{ro}_{\theta u}(s)$ (Exercise 3). Keeping in mind that the most interesting part of the Bode diagram is around the cross-over frequency, explain why the reduced-order model is sufficiently accurate for the present control purpose.

Show MATLAB figures to teaching assistant.

Exercise 6. Closed loop system

Draw the block diagram of the closed loop system (unity feedback).

The MATLAB command "`feedback(H3reduced,1)`" adds a negative feedback loop of gain 1 to the system $H^{ro}_3(s)$. Calculate the closed loop transfer function both analytically ($H^a_{cl}(s)$) and using the MATLAB command "`feedback(H3reduced,1)`" (H^m_{cl}). Compare the results.

Simulate the closed loop system (either $H^a_{cl}(s)$ or $H^m_{cl}(s)$) for a step input. Explain in terms of overshoot M_p, rise time t_r, settling time t_s and steady state error e_{ss} if the system meets our demands.

Show figures to TA. Fill in the table:

	Needed for specifications	Derived from step response
Overshoot M_p		
Rise time t_r		
Settling time t_s		
Steady state error e_{ss}		

Exercise 7. State feedback controller

For the state space model from exercise 4, design a state feedback controller. You may use MATLAB for the calculations, but be sure to keep all commands.

1. Determine the reachability matrix W_r, verify that the system is reachable, and determine W_r^{-1}.
2. Determine the characteristic polynomial $\det(sI - A) = s^2 + a_1 s + a_2$.
3. We would like to place the eigenvalues of $A - BK$ at $-50 \pm 30i$. Calculate the characteristic polynomial $s^2 + p_1 s + p_2$ for this situation.
4. Determine the state feedback gain K which places the eigenvalues of $A-BK$ at the desired location.
5. Determine the reference gain k_r.

Exercise 8. Output feedback controller

A state feedback controller assumes we can measure all the states. For this model, we can only measure x_1 (see (1)). Design an observer so we can use our state feedback controller anyway. The procedure is identical to the one you used in the fifth tutorial and follows theorem 7.2 from the book.

1. Determine the observability matrix W_o of the system, and verify that it is observable.
2. Determine the observable canonical form, and $\tilde{W}_o$.
3. Compute the observer gain L. Place both poles at -200.

Simulink

Open Simulink by typing 'simulink' at the Matlab prompt. Open the library as well by clicking the appropriate button in the toolbar.

Exercise 9. Simulink model

- In Simulink, model a State Space system as in Practicum 1 (with Gains and Integrators - do not use a State Space block[1]). Populate it with the model derived in exercise 4.
- Build a feedback loop with negative unit feedback. Also send the output to a scope ('Sinks'). As reference signal, use a Pulse Generator ('Sources') set to produce a 50% pulse width with period of 5 seconds.
- Run your model.

Show MATLAB figures to teaching assistant.

Exercise 10. Simulink with observer

The next step is adding the observer to the model. Recall that the observer dynamics are

$$\dot{\hat{x}} = (A - LC)\hat{x} + Bu + Ly.$$

- Add a state space system (again with Gains and Integrators).
- Feed u and y into it.

Exercise 11. Simulink with output feedback

We are now ready to insert the controller. This will replace the Sum block you used in the feedback loop. Use Gain blocks and a Sum block to feed $u = -K\hat{x} + k_r r$ into the Plant.

Also see how the system responds to a Step input. Does your design meet the specifications?

Exercise 12. Saturation

Having met the specifications, we have one hurdle remaining: the power supply attached to the motor you will use next week has practical limits on the amount of energy it can deliver. This changes the system from a linear system into a nonlinear one, and makes it (nearly) impossible to meet the specifications.

Simulate the effect of this limitation by inserting Saturation blocks, with upper and lower limits of ± 12 V, right before the Observer and the Plant. What do you observe?

Save your work so you can re-use it in the lab practicum!

[1] The State Space block will have the same correct results, but in the Lab practicum next week you will find that it requires Matlab to rebuild the motor program every time you make a change, whereas using gains allows you to quickly update their values without recompilation.