Experiment procedure
Robot Modelling, Identification, and Control

COD - Observer and controller design for a robot link

June 12, 2025

⚠ **IMPORTANT:** It is essential that you carry out the following steps before starting the experiment!

1.) Select "`Fixed-Step`" as the solver for your Simulink model with a variable Sample Time $T_s = 0.001$. You will select this later depending on the task. You can set this under "`Model Configuration Parameters`" in the upper bar.

2.) Avoid hardcoded values, i.e. only use variables within Simulink and define them outside in a central script which is called by the simulation via callback[1].

3.) Deactivate the check mark at "`Limit data points to...`" in Scopes in order not to lose any data points during longer simulation times.

4.) If you need to compare two systems, the easiest way is to copy the original system and make the changes to the copy. So you always have both versions available.

5.) For "`To Workspace`" blocks, select "`Array`" as the storage format, since they are the easiest to handle.

6.) If the function of a command is not clear, use MATLAB Help.

7.) Use the "`clear`" command in your main script to clean up your workspace before performing a task and avoid errors due to old data.

⚠ **IMPORTANT:** In Moodle you are provided with the MATLAB and Simulink files that you will need for the experiment. **Items marked with a ★ must be included in your experiment report**.

1 Experiment

The aim of this task is to design a continuous-time Luenberger observer for the states of the real robot link. But first we have to model our system. The dynamics of the **real** system are given in equations (1) and (2). We use these equations to obtain the **linearized** system model according to the followings.

1.) The motor voltage $u(t)$ is the system input and the motor position $\varphi(t)$ is the system output (what is measured).

2.) Use equations (1) and (2) to derive the (linearized) state space representation. The states of your linear system have to be $\varphi(t)$, $\dot{\varphi}(t)$, and $i(t)$, respectively. The system must be linearized around $\varphi_0 = 0$.

3.) Use syntax `ss` to define the linear continuous state space representation of your system in Matlab.

4.) You can use `state-space` block in Simulink to simulate the linearized system.

5.) The initial state of the system is $\boldsymbol{x}(t) = [0\,0\,0]^T$

Hint: The eigenvalues of the linearized system matrix **A** are:
$p_{1,2} = -4.548 \pm 9.676i$ and
$p_3 = -5.928$.
You can check this using command `eig`.

T1 (**3p**) Using the linearized model of the real system we can now design our observer. With the command `place`, compute the Luenberger observer gain **h**. For this use the following syntax:

```
h = place(A',C',q).';
```

The matrices **A** and **C** are those of the linearized system. The parameter **q** stands for the observer poles, which you must select **appropriately**.
Hint: Please note that that we compare the convergence rate of the observer with the one of the **linearized** continuous system.
Hint: If you intend to use gain blocks to implement your observer, please make sure that you choose the correct **multiplication** mode for the block. i.e. *Matrix* $(K * u)$ for Ax, etc.

Now test whether the output of the observer corresponds to the real system. Apply a constant voltage of $u = 1$ V to the system, the external torque is 0 Nm. You should see that the observer matches the real system almost perfectly.

★ Elaborate in your report about the process you followed to select the poles.
★ Provide in your report a figure witht the output of the system and the observer. Why does the observer match the real system almost perfectly?

T2 (**3p**) The next step is to control the system using state feedback.

a) Extend your model so that the states of your observer are fed back into the system via the gain k (will be calculated in the next steps). In this task, no external torque is applied to the link.

b) Derive the controllability matrix and check if the linearized system is controllable. You can use the command `ctrb` for this.

c) Define the symbolic variable `s` as follows

```
syms s;
```

and derive the **linearized** system characteristic polynomial.

d) use the command `sym2poly` to extract the coefficients of the characteristic polynomial to a vector $\boldsymbol{a}$.
⚠ **IMPORTANT:** please pay close attention to the order of the coefficients of the characteristic polynomial in the vector $\boldsymbol{a}$.

e) Now, as you have the coefficients of the characteristic polynomial, obtain the auxiliary matrix $\boldsymbol{W}$ and then the transformation matrix $\boldsymbol{T}$.

Hint: You can check the correctness of the transformation matrix by transforming the linearized system matrix ($\boldsymbol{A}$) to its canonical form ($\boldsymbol{A}_R$).

f) The poles of the closed-loop system should be at $p_{1,2} = -10$, $p_3 = -20$. Use the transformation matrix to obtain the feedback gain.

★ **What is the value of the gain k?**
★ **How can you check if the computed k is correct?**
★ **Apply the set point $\varphi_\text{d} = 1$ to the system, provide in your report plots of the evolution of φ of the real system and the estimated $\hat{\varphi}$ by the observer, on top of each other.**

T3 (**2p**) In this task we intend to remove the steady-state error by adding an integrator compensator to the control loop. For this, copy the previous system with the observer and add the integrator to the loop. Start with small gains (< 1) for the integrator. Change the gain such that the steady-state error becomes zero.

★ **Include a screenshot of the block diagram of the whole system with feedback and observer in your report.**
★ **Plot the estimated and real angle φ in your report.**

In order to create the control error for the integrator compensator, you can either feedback the real angle φ or the estimated one $\hat{\varphi}$.
★ **Which one did you choose and why?.**

T4 (**2p**) In the last task, we intend to estimate a constant disturbance applied to the system. The disturbance appears as an external torque τ_ext in our system:

$$u(t) = Ri(t) + L\dot{i}(t) + k_e\dot{\varphi}(t)$$
$$(J + ml^2)\ddot{\varphi}(t) = k_m i(t) - d\dot{\varphi}(t) - mlgsin(\varphi) + \tau_\text{ext}$$

a) Copy and paste only the robot link, without any observer or feedback. The system input is 0 in this task.

b) Add a constant block with the value of 0.2 (Nm) and connect it to the external torque input of the robot link. Now, 0.2 Nm disturbance is applied to your system.

⚠ **IMPORTANT:** If you use `place` syntax for computing the observer gains, one component of the observer gain becomes zero. Play around with the corresponding gain until the observer follows the real system.

★ **How do you re-design your observer such that the external torque is also estimated? Write down the new linearized system matrix.**
★ **Now design the observer. Include in your report plots the estimated external torque and the real one.**

Equations

$$u(t) = Ri(t) + L\dot{i}(t) + k_e\dot{\varphi}(t) \tag{1}$$

$$(J + ml^2)\ddot{\varphi}(t) = k_m i(t) - d\dot{\varphi}(t) - mlgsin(\varphi) \tag{2}$$

Parameter

Parameter	Description	Value	Unit
J	motor moment of inertia	0.001	Kg m^2
d	Damping constant	0.1	Nm s
k_e	electrical motor constant	0.1	$\frac{\text{V s}}{\text{rad}}$
k_m	mechanical motor constant	0.1	$\frac{\text{Nm}}{\text{A}}$
L	Inductance	0.01	H
R	electrical resistance	0.1	Ω
m	mass of the link	1	Kg
l	lever arm of the link mass	0.1375	m s

Table 1: Values for the pneumatic motor

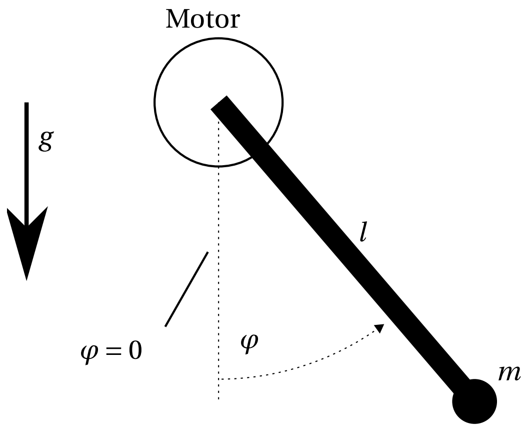

Figure 1: Sketch of the robot link.