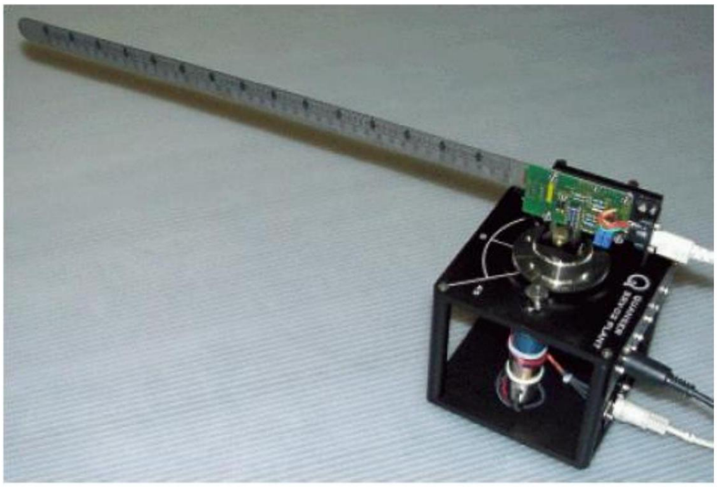
Control of Cyber-physical Systems

LABORATORY WORK REPORT — PART II

Computer Control of a Flexible Robot Arm Joint



Shift L03

Group 10:
André Teixeira — 99891
Rodrigo Andrade — 100074
Tiago Costa — 100096

2024/2025 – 1st Quarter

Instituto Superior Técnico
Department of Electrical and Computer Engineering
Scientific Area of Systems, Decision, and Control

Contents

Introduction

This section focuses on designing and implementing a control system for a flexible robot arm joint, using a state-space model developed earlier. The goal is to create a system that can accurately position the tip of the flexible bar. The controller is designed using Linear Quadratic (LQ) techniques, which help optimize performance by balancing control precision with energy efficiency.

Since the plant's state can't be measured directly, a Kalman filter is used to estimate the necessary values. The Loop Transfer Recovery (LTR) technique is then applied to make the system more robust and resistant to disturbances.

The controller is tested on the physical system, and its performance is analyzed to ensure that it meets the design expectations. The results are documented with MATLAB scripts, Simulink diagrams, and a detailed report to demonstrate how the system behaves under different conditions.

1 Question 4 - Controller design and testing of the controlled system (17 pts)

1.1 Insert the MATLAB scripts, comments included. (1.5 pts)

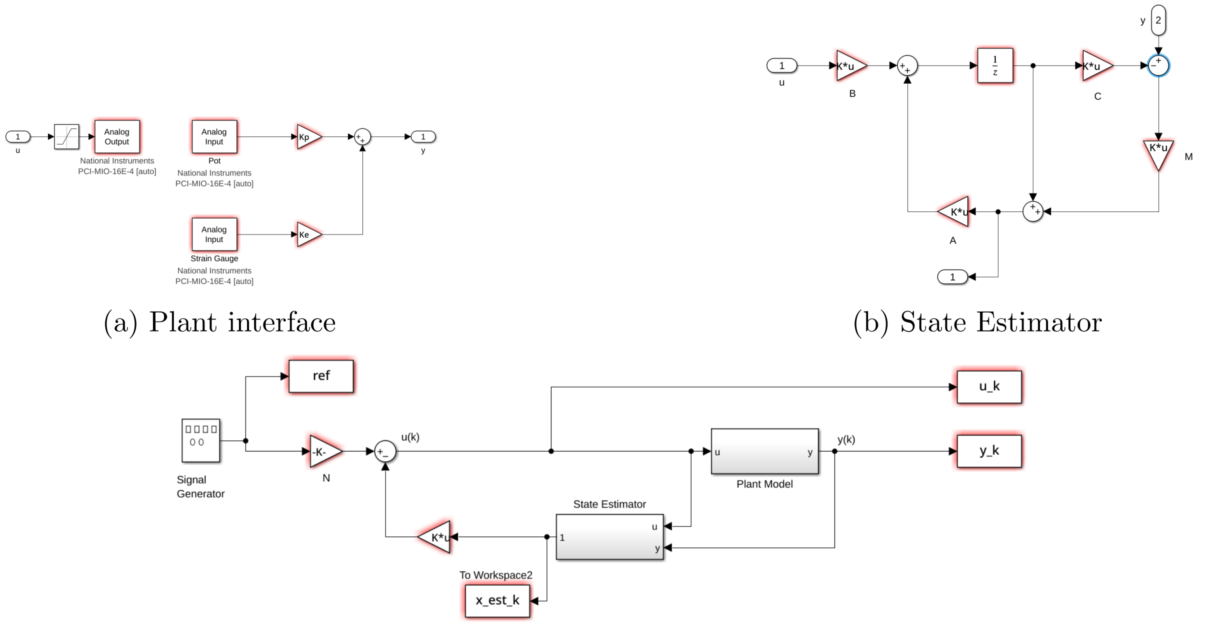
Below is shown the MATLAB script used to perform the simulation of the system. The simulation was carried for multiple values of Q_e, R_e and R in order to analyze the system's behavior and achieve the best trade-off between performance and stability, allowing us to identify the most suitable configuration for the system's control.

```matlab
% Clear workspace and load the necessary data
clear; clc; load('IdentificationData.mat')

%% Define system parameters
Q = C'*C; R_ = 10; G = B; Qe_ = 1; Re_ = 1;

%% Compute the optimal gain for the state feedback controller
[K,S,e] = dlqr(A,B,Q,R_);
Nbar = inv(C*inv(eye(size(A)) - A + B*K)*B);
%% Compute the optimal gain for the state estimator
[M,P,Z,E] = dlqe(A,G,C,Qe_,Re_);
time = 0:sampling_time:120;

%% Run the simulation
open_system('parte2_simulink.slx');
set_param('parte2_simulink', 'SimulationCommand', 'start');
sim_out= sim('parte2_simulink.slx');

%% Save the simulation results
filename = [datestr(now, 'dd-mm-yyyy-HH-MM-SS') sprintf('
    _parte2_R%d_Qe%d_Re%d_GI0_0175_noise.mat', R_, Qe_, Re_)];
save(filename);
```

1.2 Show and describe the block diagram used to control the system. (1.5 pts)

All simulink models devloped in this part of the course can be seen in this chapter. The model in Figure 1a represents the plant interface, facilitating the interaction between the physical system and the control model, specifically, sensor data and actuator signals are processed through this interface. The model present in figure 1b depicts the state estimator, which is responsible for estimating the system states based on sensor inputs and control signals, utilizing a Kalman filter. Lastly, the state feedback controller, shown in Figure 1c, is tasked with generating the control signals sent to the actuators. This controller does not incorporate integral action, this means that if the motor or gearbox has any type of non moduled dead zones the system will have a steady state error.

To fix this possible problem a new optimal controller with integral action was designed, as we can see in figure 2, this new control simply integrates the error of the system as sum to the control signal.



(a) Plant interface

(b) State Estimator

(c) Controller without integrator

Figure 1: Simulink models utilized for system control without integral action

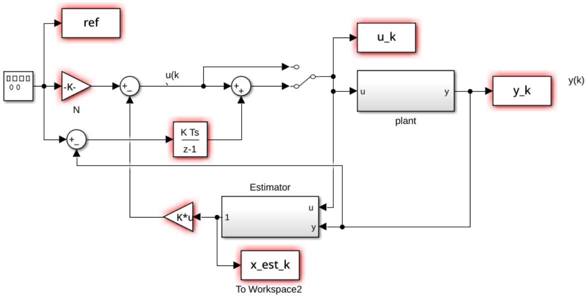


Figure 2: Controller with integrator

1.3 Comment on the choice of the weights on the quadratic cost when using the LQG design approach. Include the root-square locus. (3 pts)

The cost function being minimized in our system is defined as shown in equation 1, where Q and R represent the weights of the quadratic cost. By adjusting these values, we effectively control the relative importance of the system's state versus the control signal. This allows us to prioritize either minimizing the state error, irrespective of the energy consumed (control signal), or reducing energy usage while accepting a larger error.

Given the state space formulation and the specific characteristics of our system—namely, that the control signal does not have an immediate effect on the output (i.e., D = 0)—we must set $Q = C^T C$. This reduces the problem to finding the optimal value for R that yields the best performance.

To determine the appropriate value of R, we begin by examining the square root locus, as shown in figure 3. At first glance, the presence of poles outside the unit circle suggests an unstable discrete-time system. However, these poles actually belong to the closed-loop system, i.e., they are the poles of $1 + G(z)G(z^{-1})$. Therefore, we are only concerned with the poles inside the unit circle, as the poles outside it are due to $G(z^{-1})$ and do not have any physical significance. Inside the unit circle, we observe a set of poles (marked in red) corresponding to the estimator. Since the controller cannot be faster than the estimator, this provides an upper bound for R, which, in this case, is determined by the inverse of the estimator gain—set to 1.

From this analysis, we applied a trial-and-error approach, testing multiple values of R. The system's response to these variations is shown in figure 4. The optimal result was achieved with $R = 50$, as it minimized the delay seen with smaller values of R, while still reducing the system's energy consumption. It may be important for the reader to note that the system response in figure 4 uses the controller present in figure 2, which incorporates integral action with saturation limits, the reason for this will be explained in a later question.

$$J = \sum_{k=0}^{\infty} \left[(x^T(k)Qx(k) + u^T(k)Ru(k) \right] \tag{1}$$

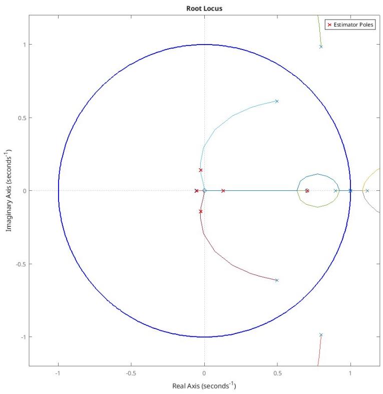


Figure 3: Root Locus

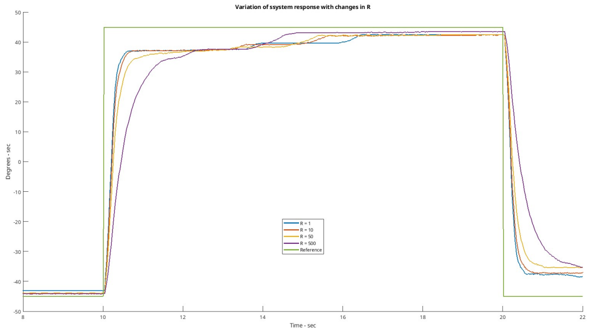


Figure 4: System Response variation with R

1.4 Explain the effect of the choice of the noise covariance matrices in a LTR framework; (3 pts)

In a LTR framework, the choice of noise covariance matrices plays a crucial role in the performance of the system. The process noise covariance matrix Q defines the uncertainty associated to the state dynamics of the system and the choice of a larger Q suggests that the system is being exposed to more significant disturbances in the model. This means that the Kalman filter will value more the measurements performed by the sensors, potentially leading to faster adjustments in the controller. On the other hand, the measurement noise covariance matrix R gives us the uncertainty in the measurements obtained from the sensors and the choice of a larger R suggests more noise in the measurements. This means that, in this case, the Kalman filter will trust more the predictions produced by the model than the measurements.

1.5 Display and analyze the resulting closed-loop frequency response and time response. (3 pts)

1.6 Comment on effect of the inclusion of a pre-filter. (2 pts)

Pre-filtering can play a vital role in shaping the reference signal before it enters the feedback loop by adjusting the input signal to compensate any new dynamics introduced by the controller or the plant.

One of filters present in our system was an integrator, which was used to eliminate the steady-state error and push the system toward the desired output, this was particularly useful since the dead zone of the motor and gearbox introduced a non modulated error, that mean for a small value of the actuaction signal, was noticed to be around 0.2V, the system would not move as we see in figure 5. Something important to note in our integral is we have saturation limits, this was done to avoid any type of integral windup that could have happened. These saturation limits in our system were set to -2.5V and 2.5V, since the with the value of R choosen the maximum optimal control signal that we seen was around 5V, so we decided in a saturation of 2.5V since it would make sure the integral could never overwrite the maximum optimal control signal.

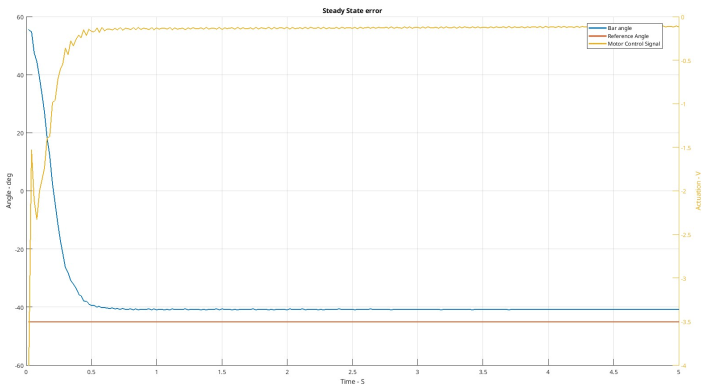


Figure 5: steady State Error

Another pre-filter present in our system was the kalman filter, and this is a rather important was, since as discussed earlier allows us to estimate the state of the system from the output, the control action and the model.

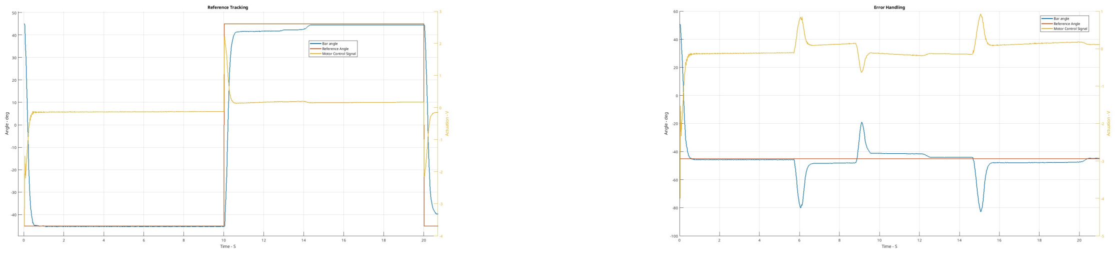
1.7 Discuss how do you evaluate the performance of your control system and what are the limits of performance. (3 pts)

Our objective for this part of the course was to be able to control the position of the tip of a robotic arm. This came with some chalenges, the first one being that the object used to represent the arm, was a thin metal ruller that was not very rigid, meaning oscilation would happen whenever we moved, the second was the existance of dead zone in the motor and gearbox, this dead zone meant that for small values of the control signal the system would not react, meaning that the system would not reach the desired position, but would stop close to it, and the third main problem we had to deal with was the fact that we were trying to control a linearized version of our system, but the real system was not a linear one, this would induce errors in our model, and would make the system harder to control.

Knowing this we can define two metrics for evaluating the performance of our control system, the first one is how accurate and how fast we can track a desired reference and the second one is how much we can handle big disturbances, such as an object coliding with the arm. Other metrics could be used depending on the application, for example, for a robot where energy is limited, it may be more important to minimize the energy used to reach a position rather than how accurate we track said position.

The limits of performance of our control system are mainly non linear effects that are not taken into consideration in our model, such as the lack of rigidity from the arm, or the dead zone in the motor and gearbox. Also since we are using a discrete time controller, we will also have the problem of only controling the system in certain time steps, meaning that is for some reason we acheive an oscilation of aroungd 50Hz, our sampling rate, we would not be able to detecte it, meaning we would not stop it. Also, there is a limit for the speed of the system, since the both the motor and the arm have a maximum speed, the motor due to the voltage safety limits, and the arm due to the lack of rigidity and the air resistance.

After considering how we could evaluate our system and its performance limits, we now present the plots from Figure 6. On the left, Figure 6a, the system response accurately



(a) Reference Tracking

(b) Disturbance handling

Figure 6: Plots showing the performance of the control system

tracks the reference, requiring approximately 4 seconds to follow a 90-degree change in reference. Notably, the first 96% of the change is achieved within the first 0.5 seconds, which is an excellent result. The system stops initially at 41.6 degrees, very close to the target of 45 degrees.

On the right, Figure 6b, we observe the system's response to multiple disturbances (three in total). The first disturbance occurs slightly after the 5-second mark, and the system returns within 5 degrees of the target in under a second after releasing the arm. However, the reference error is not fully eliminated as a new disturbance occurs at the 9-second mark. In both this disturbance and the next, at 15 seconds, the system again reduces the deviation to within 5 degrees of the target in about 0.5 seconds. Furthermore, as observed earlier, it takes approximately 3.5 seconds after reaching this plateau for the system to achieve the reference target position.

Conclusions

This project successfully achieved the design and implementation of a control system for a flexible robot arm joint. By applying Linear Quadratic Gaussian (LQG) control and Kalman filtering, the system was able to precisely control the positioning of the flexible bar, while maintaining stability and addressing uncertainties.

Through testing and validation, the controller demonstrated effective performance, meeting the required criteria in both time and frequency domains. The use of Loop Transfer Recovery (LTR) further improved the system's robustness, ensuring that it could handle disturbances.

Overall, this work illustrates the practical use of advanced control techniques in managing cyber-physical systems. The documentation, along with the MATLAB scripts and Simulink models, offers a solid basis for future enhancements and further development.