Lab 3 Report: Position control

Marcos, Laia, Claudia & Mariona

December 14, 2025

Universitat Politècnica de Catalunya

1 Introduction

In this lab, we worked on the position control of an exoskeleton joint. The main goal was to see if the motor could accurately follow a specific movement path, known as a trajectory.

First, we wrote a script to generate a reference signal (a sine wave) to tell the exoskeleton where to move. We then recorded how the actual device responded to see if there was any delay or error. Finally, we performed a system identification test. By sending sine waves at different speeds (frequencies) to the motor, we could build a Bode plot and figure out the system's bandwidth—basically, how fast it can move before it starts lagging too much.

2 Reference Generation

2.1 Sine Wave Generation

In this exercise, a specific sinusoidal reference signal was created to test the system. The requirements were an offset of 45°, an amplitude of 20°, and a frequency of 5 rad/s[cite: 38].

The following MATLAB function was implemented to generate this signal:

```
function signal = ReferenceGenerator(t, A, w, phase)
    % Generate signal with Amplitude A, frequency w, and phase offset
    signal = A * sin(w*t + deg2rad(phase));
end

% Parameters used in Lab 3
A = 20;                     % Amplitude
w = 5;                      % Frequency (rad/s)
phase = 45;                 % Offset
```

Listing 1: MATLAB function for Reference Generation

2.2 Time Response Analysis

The system was excited with the generated signal, and the actual position was recorded. The goal was to plot the reference position versus the actual position to evaluate the sensor performance and tracking accuracy[cite: 41].

The plotting process utilized the following code:

```
% Load data
[desired_pos, actual_pos, t_sim] = load_recording('Lab3_2.1.txt');

% Plot Reference vs Actual
figure;
subplot(2,1,1);
plot(t_sim, desired_pos, 'LineWidth', 1.5); hold on;
plot(t_sim, actual_pos, 'LineWidth', 1.5);
legend('Reference', 'Actual');
xlabel('Time [s]'); ylabel('Position [deg]');

% Plot Error
subplot(2,1,2);
```

```
plot(t_sim, desired_pos - actual_pos);
ylabel('Error [deg]');
```

Listing 2: Code for plotting Time Response

Figure 1 shows the resulting response. The actual position (orange) follows the general shape of the reference (blue), but significant artifacts are present where the position drops sharply. The position error peaks between 20° and 40°, indicating that while the sensor is functional, the tracking performance suffers from mechanical or control-related disturbances.

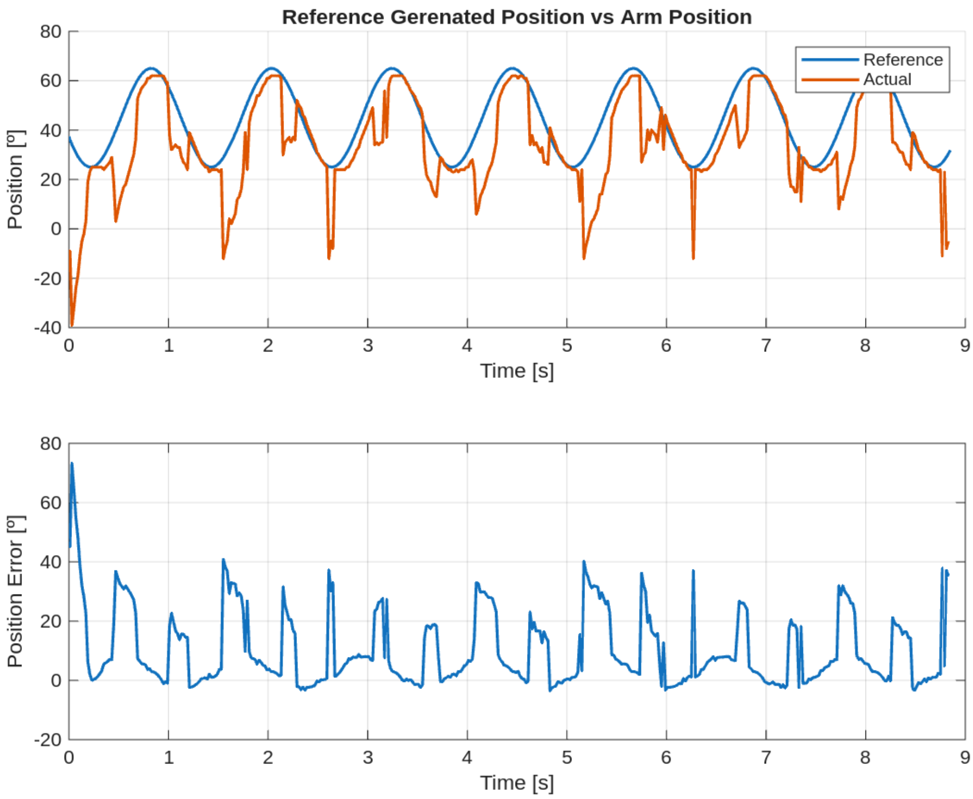

Figure 1: Reference vs. Actual Position (Top) and Position Error (Bottom).

3 System Identification in the Frequency Domain

The exoskeleton was approximated as a linear system $G(s)$[cite: 59]. The identification process involved exciting the system at different frequencies to extract the frequency response[cite: 62].

3.1 Experimental Bode Plot

The system was excited with frequencies ranging from 0.5 to 9 rad/s[cite: 94]. For each recording, the magnitude ratio (gain) and time delay (phase shift) were extracted automatically to construct the experimental Bode plot.

```
% Loop through recorded files for different frequencies
for i = 1:length(files)
    [des, act, t_s] = load_recording(files{i});
```

```

    % 1. Estimate Frequency (w) via zero-crossings/peaks
    % ... (Peak detection code) ...
    w_exp(i) = w_val;

    % 2. Estimate Gain (Ratio of RMS amplitudes)
    amp_in = rms(des_ac) * sqrt(2);
    amp_out = rms(act_ac) * sqrt(2);
    gain_exp(i) = amp_out / amp_in;

    % 3. Estimate Phase (Time delay via cross-correlation)
    [acor, lag] = xcorr(act_ac, des_ac);
    timeDiff = lag(I) * mean(diff(t_s));
    phase_exp(i) = -w_val * timeDiff * (180/pi);
end

% Plotting the Experimental Bode
semilogx(rads2hz*w_exp, 20*log10(gain_exp), 'r*-');
semilogx(rads2hz*w_exp, phase_exp, 'r*-');
```

Listing 3: Automated Parameter Extraction for Bode Plot

As seen in Figure 2, the magnitude response is unexpectedly high (> 12 dB), suggesting the output amplitude exceeded the input significantly. The phase response shows a characteristic lag that increases with frequency.

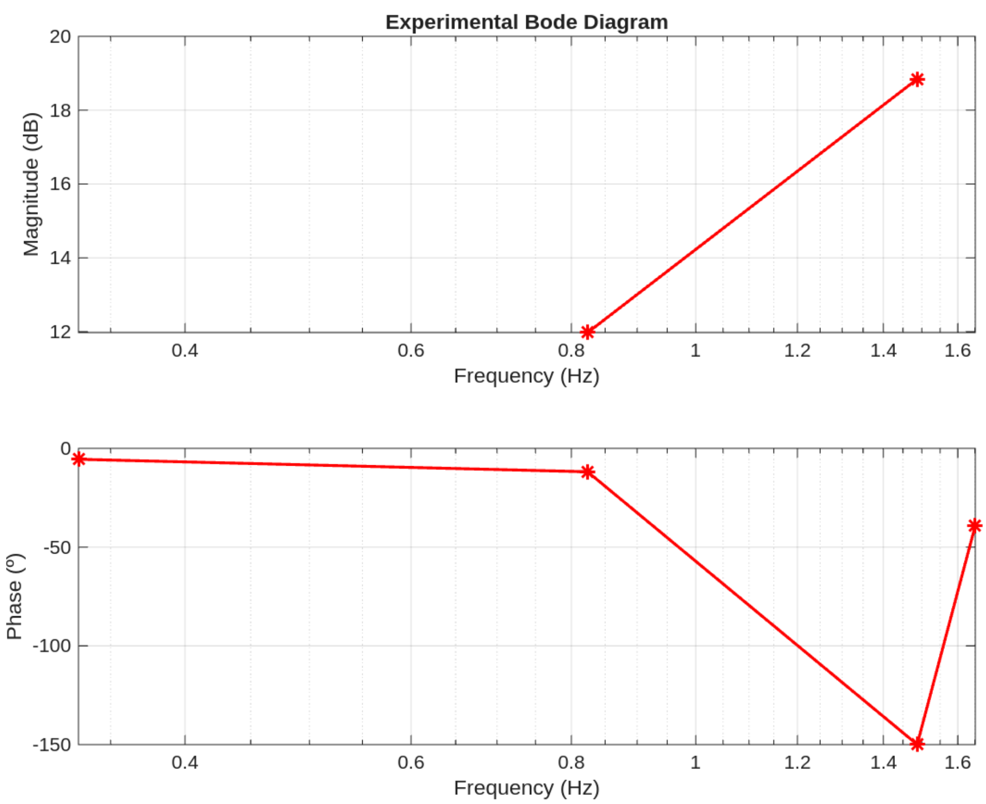

Figure 2: Experimental Bode diagram.

3.2 Model Fitting

A first-order transfer function model was proposed to fit the experimental data[cite: 113]:

$$G(s) = \frac{K}{\tau s + 1} \tag{1}$$

The theoretical parameters chosen were $K = 1$ and $\tau = 0.1$ s.

```
% Theoretical parameters
K = 1;
tau = 0.1;
num = K; den = [tau, 1];

% Generate Theoretical Bode
w_theoretical = logspace(log10(0.1), log10(100), 100);
[gain_theo, phase_theo] = bode(num, den, w_theoretical);

% Compare with Experimental Data
semilogx(rads2hz*w_exp, 20*log10(gain_exp), 'r*', ...
         rads2hz*w_theoretical, 20*log10(gain_theo), 'b-');
```

Listing 4: Model Fitting Code

Figure 3 compares the model with the experimental data. The theoretical DC gain ($K = 1 \rightarrow 0$ dB) is much lower than the experimental gain, indicating K must be increased. The phase trend is similar, though the experimental system exhibits a sharper phase drop, hinting at potential time delays not captured by a simple first-order model.

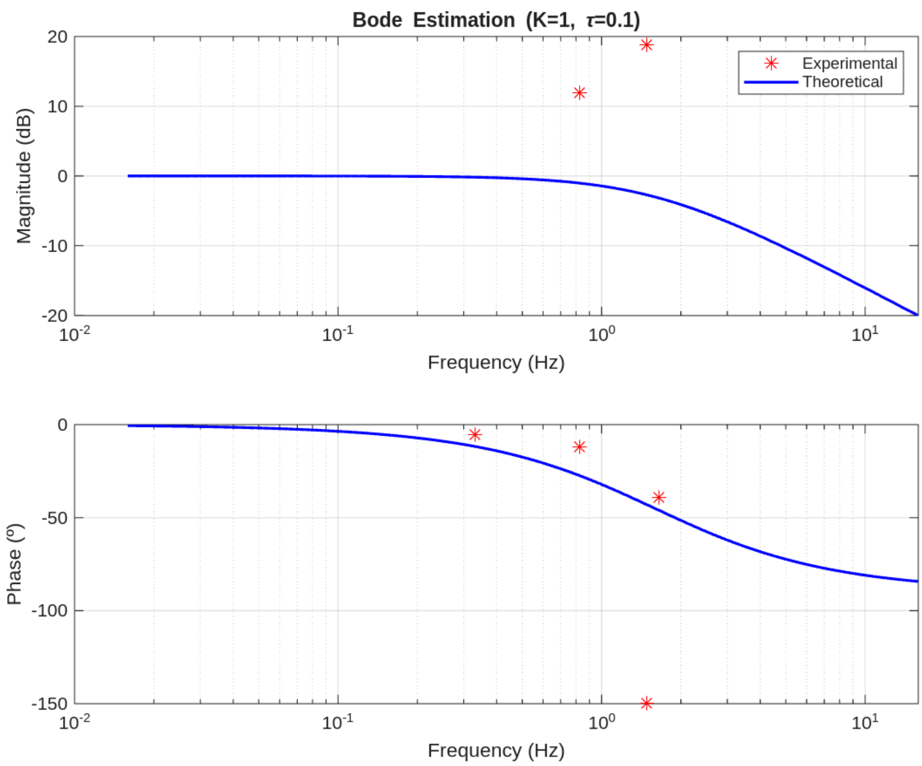

Figure 3: Comparison of Experimental Data and Theoretical Model ($K = 1, \tau = 0.1$).

3.3 Bandwidth Analysis

The bandwidth is defined as the frequency where the magnitude drops 3 dB below the DC value[cite: 65]. For the estimated first-order system ($\tau = 0.1$ s), the bandwidth is calculated as:

```
% Bandwidth for 1st order system is 1/tau [rad/s]
bw_rad = 1/tau;           % 10 rad/s
bw_hz = bw_rad / (2*pi); % ~1.59 Hz
```

Listing 5: Bandwidth Calculation

The calculated bandwidth is **1.59 Hz**. **Analysis:** While this covers the primary frequency range of normal human motion (< 1 Hz), it is on the lower side for high-performance assistive devices. A higher bandwidth (e.g., 4-8 Hz) would be preferable to handle faster reflex movements or tremor suppression effectively without inducing noticeable lag for the user.

4 Conclusion

The experiments showed that the exoskeleton can follow a path, but it isn't perfect. In the time response test, we saw that the general motion was correct, but there were some large downward spikes in the data. This looks like a sensor issue or a loose connection rather than a problem with the control logic itself.

When we looked at the frequency response (Bode plot), we noticed the real system had a much higher gain than our theoretical model assumed ($K = 1$). The phase lag also dropped off faster than expected at higher frequencies, suggesting the real hardware has extra delays we didn't account for.

Finally, we calculated a bandwidth of about 1.6 Hz. This is fine for slow, voluntary movements, but it would be too slow to handle fast reflexes or tremors. If we wanted to use this for more dynamic tasks, we would need to tune the controller to respond faster.