



EEE4118F: Process Control and Instrumentation

Robust Digital Position Control

GA Report

Safiya Mia

Student Number: MXXSAF002

Partner: Imaan Shaik [SHKIMA004]

Group Number: 21

Mechatronics Division
Department of Electrical Engineering

May 2, 2025

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.



Safiya Mia

May 2, 2025

Date

Contents

1	Purpose of the Project	1
2	Summary of the Modelling and System ID Work	1
2.1	Time Domain Identification	1
2.2	Scaling Constant	2
2.3	Frequency Domain Analysis	2
2.4	Robustness Analysis	2
2.5	Model Validation	2
2.6	Final Plant Model	2
3	Digital Robust Cascaded Controller Design	3
3.1	Controlling the Inner Velocity Loop	3
3.2	Satisfying Robustness and Frequency-Domain Specifications	5
3.3	Dealing with Integrator-Induced Phase Delay	5
3.4	Implementing the Control System in Discrete Time	6
3.5	Controlling the Outer Position Loop	6
4	Digital Controller Implementation	7
4.1	Implementation of Discrete Controller Algorithm in C#	7
4.2	Creating the Velocity Controller Model on Simulink	8
4.3	Controller Anti-Windup and Saturation Handling	8
4.4	Clamping Logic	9
5	Validation	10
5.1	Simulink Responses and Analysis	10
5.2	Lab Demonstration Responses and Analysis	12
5.3	Closing the Triangle of Theory, Simulation and Measurement	12
6	Conclusion	13
7	Appendix	15

1 Purpose of the Project

This project set out to create a robust digital position control system for a servo motor. The aim was twofold: to achieve precise position tracking and to preserve system stability even when physical parameters, such as inertia, friction, or amplifier gain, shift unexpectedly.

To reach these goals, the work unfolded in three key stages:

1. System Modelling and Identification

- Establishing a dynamic representation of the motor
- Extracting parameters through various system response tests and analysing the resulting data

2. Digital Robust Position Controller Design

- Defining performance requirements
- Designing a discrete-time controller resilient to parameter variations

3. Digital Controller Implementation

- Translating the design into code
- Testing in simulation on Simulink, and on the actual hardware in the lab

This report focuses on the aforementioned stages. It details how a trustworthy model was derived, how the system's performance targets were assessed, and how a discrete-time control strategy was formulated and implemented. Both the simulations and the laboratory tests aim to confirm that our control approach meets the accuracy and stability criteria, across a range of operating conditions.

2 Summary of the Modelling and System ID Work

The goal of Lab 1 was to identify a mathematical model of the servo system that accurately represents its dynamic behaviour under different operating conditions. This model would serve as the foundation for robust control system design and analysis.

2.1 Time Domain Identification

Identification of the servo's dynamics combined time domain and frequency domain experiments via MATLAB's System ID Toolbox. A ramp response test was conducted and the results established a linear input range of approximately 0.2 V to 0.3 V. Within this window, a step excitation of 0.1 V produced the following response that conformed closely to a first-order system:

- System Gain: $A = 49.21$
- Time Constant: $\tau = 1.6 \text{ s}$
- Identified transfer function:

$$P(s) = \frac{49.21}{1 + 1.6s}$$

2.2 Scaling Constant

To convert the position signal into velocity, a scaling factor $a = 8$ was calculated. This was done by measuring the slope of the position response during constant-speed motion and comparing it to the known velocity, ensuring consistent unit mapping between measured signals and model variables (See Figure 15 for validation).

2.3 Frequency Domain Analysis

Using sinusoidal input signals and Bode plots, the system's frequency response was analysed. The gain and phase characteristics confirmed a first-order model, with:

- Approximate gain: $A \approx 50$
- Crossover frequency: $\omega_c = 0.625 \text{ rad/s}$

This confirmed the parameters identified in the time domain and validated the system's response over a range of frequencies.

2.4 Robustness Analysis

To assess system variation, two extreme conditions were tested:

- **Brakes engaged:** Increased friction caused lower gain and faster response.

$$A = 12.36, \quad \tau = 0.118 \text{ s}$$

- **Added inertia (mass disc):** Heavier load led to higher gain and slower response.

$$A = 50.22, \quad \tau = 9.6 \text{ s}$$

These boundary cases defined the full range of expected dynamics and were used to ensure the future controller design would remain robust.

2.5 Model Validation

The identified model was validated by comparing simulated and measured step responses for all three cases: nominal (see Figure 12), brake-engaged, and added mass. The match was strong in each case. Bode plot comparisons also confirmed close alignment of magnitude and phase across frequencies (Figures 13 and 14). Minor discrepancies were attributed to measurement noise and resolution limits.

2.6 Final Plant Model

The nominal system transfer function was confirmed to be:

$$P(s) = \frac{49.21}{1 + 1.6s}$$

The parameter ranges for robustness were:

$$12.36 \leq A \leq 50.22, \quad 0.118 \text{ s} \leq \tau \leq 9.6 \text{ s}$$

The consistency across identification methods, simulation results, and experimental data confirms that the model is reliable for subsequent robust control design and implementation.

3 Digital Robust Cascaded Controller Design

The intention behind the design of the servo motor controller was to ensure perfect position tracking and robustness. The design was guided by a **cascaded control** structure, consisting of an inner velocity loop nested within an outer position loop. The design process involved defining performance criteria, designing the controller in the continuous domain, discretising it for digital implementation, and validating its performance via both simulation and experimental testing. The aforementioned elements of the design process will be documented in further detail in this section.

3.1 Controlling the Inner Velocity Loop

A Proportional–Integral controller was adopted to regulate the speed loop to guarantee accurate position tracking and satisfy strict time–domain requirements. This choice delivers rapid settling, controlled overshoot, and elimination of steady–state error—essential characteristics for high-performance servo operation.

The identified nominal plant

$$P_{\text{nominal}}(s) = \frac{49.21}{1 + 1.6 s}$$

captures the motor’s first-order velocity behaviour and serves as the foundation for tuning.

Category	Details
Time-Domain Specifications	<ul style="list-style-type: none"> • Closed-loop poles left of $\Re\{s\} = -1$ (settling time < 1 s) • Maximum overshoot $\leq 20\%$ ($\zeta \approx 0.46$) • Zero steady-state error for step inputs (integral action) • Strong disturbance rejection with minimal transient deviation
Initial Gain Selection (from Root Locus)	Root-locus analysis yielded $K_p = 1$ and $K_i = 0.7$, positioning poles within the desired region.

Table 1: Summary of Controller Requirements and Continuous-Time Design

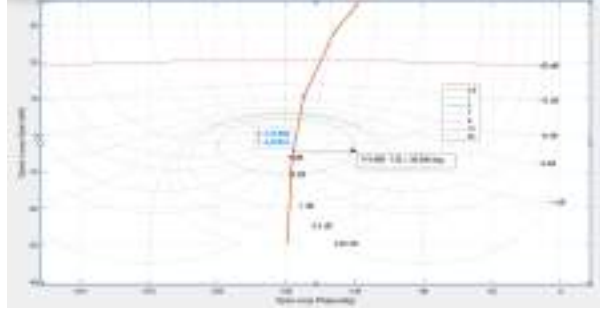


Figure 1: Inverse Nichols Chart of the Nominal Plant

The Inverse Nichols chart in Figure 1, shows intersection with the 3 dB robustness contour at $\omega = 7$ rad/s, indicating a sensitivity peak.

To mitigate this peak, a phase lead term was required. The uncompensated phase deficit was

$$\Delta\phi = 180^\circ - 135^\circ = 45^\circ,$$

and the digital sampler ($T = 0.03$ s) introduces 10.2° lag, so the lead compensator must supply

$$45^\circ + 10.2^\circ = 55.2^\circ.$$

Using the standard lead design relation,

$$\alpha = \frac{\omega}{\tan(\phi_{\text{lead}})} = \frac{7}{\tan(55.2^\circ)} \approx 8.9,$$

the zero and pole were placed at $s = -7$ and $s = -8.9$, respectively.

Combining the PI and lead elements produced the final regulator

$$G_{\text{PI}}(s) = 8.2 \frac{s/8.9 + 1}{s},$$

which after fine-tuning yielded

$$K_p = 0.828, \quad K_i = 9.2.$$

This design meets transient and steady-state requirements while significantly improving speed-loop robustness against model uncertainties.

3.2 Satisfying Robustness and Frequency-Domain Specifications

Robustness targets were cast into the frequency domain through Quantitative Feedback Theory. The sensitivity function

$$S(j\omega) = \frac{1}{1 + L(j\omega)}$$

was required to satisfy

$$|S(j\omega)| \leq 0.1 \quad (-20 \text{ dB}) \quad \text{for } \omega \ll 1,$$

thereby limiting the effect of disturbances and model uncertainty at low frequencies.

A collection of plant models spanning the identified uncertainty range was generated at key frequencies and plotted as “templates” using MATLAB’s QFT Toolbox. These templates were superimposed on the QFT M -circle constraints to confirm that every prospective loop shape remained inside the admissible region.

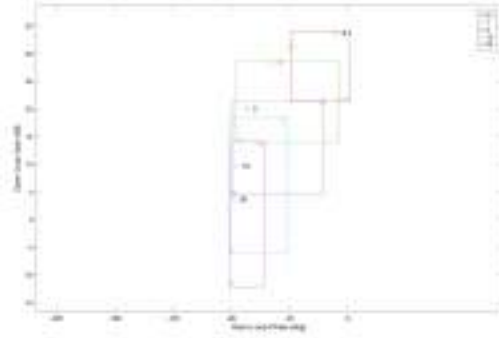


Figure 2: Plant Templates plotted using QFT Toolbox

After evaluating the trade-offs between speed and stability margins, a nominal loop gain of $K \approx 9.2$ was selected. This design was then tested under the most extreme plant conditions in Simulink and validated on the hardware in the lab to ensure compliance with the robustness specification.

3.3 Dealing with Integrator-Induced Phase Delay

Adding an integrator to the speed loop guarantees zero steady-state error for step commands, upgrading the system to Type 1, but unavoidably imposes a 90° phase lag. This extra lag erodes phase margin and risks breaching robustness targets.

Simulation results confirmed that a bare integrator caused violations of both the low-frequency sensitivity bound and the overall robustness envelope. To address this, a lead

compensator was inserted (as detailed previously) and its parameters were adjusted until the loop's frequency response fell back within the specified limits.

By blending classical time-domain criteria (settling time, overshoot etc.) with frequency-domain robustness analyses (QFT templates and Nichols diagrams), the final controller meets all performance and stability requirements under the full spectrum of modelled plant variations.

3.4 Implementing the Control System in Discrete Time

To enable digital control, the continuous-time PI controller was converted into discrete form using the bilinear approximation, with a sampling interval of $T = 0.1$ seconds:

$$s \approx \frac{2(z-1)}{T(z+1)}$$

Substituting this into the analogue controller expression produced the following discrete-time implementation:

$$C(z) = \frac{0.828z - 0.69}{z - 1}$$

This discrete controller was modelled in Simulink and tested against all identified plant variations. Frequency-domain analysis using Nichols charts demonstrated that the resulting closed-loop system maintained sufficient phase margin and appropriate gain crossover frequency. The inclusion of phase lead successfully repositioned the frequency response away from sensitive regions, as confirmed in both continuous- and discrete-domain analyses.

The implemented difference equation, reflecting the final digital control law, is expressed as:

$$u_{i+1} = 0.828 u_i - 0.69 e_{i+1} \quad (1)$$

3.5 Controlling the Outer Position Loop

A proportional control strategy was adopted in the outer loop to regulate position. The selected gain value of 0.6 was tuned to ensure precise tracking while avoiding actuator saturation. Position was derived by integrating the velocity signal produced by the inner speed control loop, and the resulting signal was compared to the reference input to generate the appropriate corrective action.

Simulations of the full cascaded system demonstrated effective tracking behaviour, with low overshoot and prompt rejection of disturbances. These outcomes held consistently across all tested plant configurations, including the nominal case, increased inertia, and heightened friction. The system also preserved stability and performance despite the influence of discretisation and minor non-linear dynamics.

4 Digital Controller Implementation

During experimental implementation, slight tuning of the controller gains was required to address actuator saturation and mitigate overshoot caused by non-idealities in the hardware. To counteract integral windup, anti-windup logic and a gain-limiting mechanism were incorporated into the control loop. These modifications ensured that the real-time behaviour closely matched simulation results.

4.1 Implementation of Discrete Controller Algorithm in C#

This algorithm determines the control signal at each timestep using the current setpoint, measured output, and velocity feedback. The table below details the role of each step in the computation.

```
1 //Controller
2 double Error = rt_SetPoint - yt_PlantOutput;
3
4 double Error1 = Gain * Error;
5
6 double Error2 = Error1 - Velocity;
7
8 double ut_Unsat = Un_1 + 0.828 * Error2 - 0.69 * En_1;
9
10 // Send output to DAQ
```

Listing 1: C# Code Implementing the Control Algorithm

Line Number in Code	Explanation
Line 2	Computes the error by finding the difference between the setpoint and the actual output. This value shows how much correction the controller must apply.
Line 4	Multiplies the error by the proportional gain Gain , providing an immediate corrective signal proportional to how large the error is.
Line 6	Incorporates velocity feedback by subtracting the measured speed, which reduces rapid responses and improves system stability.
Line 8	Executes a discrete PI control update using backward Euler integration. The new control action combines the influence of both past and present error values, scaled by tuned gains.

Table 2: Breakdown of the C# Controller Algorithm

4.2 Creating the Velocity Controller Model on Simulink

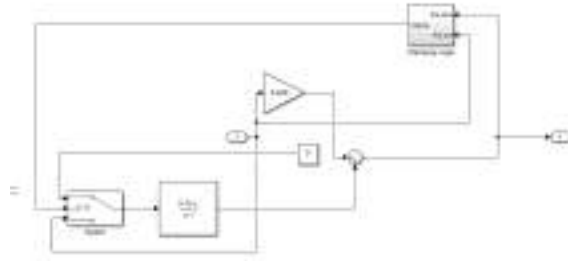


Figure 3: Simulink Model of Velocity Controller

- **Summation Block:**
 - Combines the proportional and integral terms to compute the final control output $u[k]$.
- **Proportional Gain:**
 - Applies the proportional gain $K_p = 0.828$ to the velocity error, generating the proportional component of the control signal.
- **Discrete Integrator:**
 - Uses the backward Euler method to accumulate the error over time, given by $\frac{K_i T_s z}{z-1}$, with $K_i = 9.2$ to integrate the error signal.
- **Switch Block:**
 - Detects saturation conditions and, when triggered, prevents further integration by setting the input to zero, effectively halting error accumulation and avoiding windup.
- **Clamping Logic Subsystem:**
 - Continuously monitors the system for saturation. When detected, it outputs a boolean signal (Clamp), which controls the switch and disables integration.
- **Error Calculation:**
 - Computes the velocity error as $e[k] = r[k] - y[k]$, where $r[k]$ is the reference signal and $y[k]$ is the actual system output.

4.3 Controller Anti-Windup and Saturation Handling

```

1 // Anti windup clamping
2 if (ut_Output == ut_Unsat)
3     DAQ.Output(0, ut_Output);
4
5 // ASSIGN PREVIOUS VALUES
6 Un_1 = ut_Output;
7 En_1 = Error2;

```

Listing 2: Anti-Windup and State Update

Line Number	Explanation
Line 2	Checks whether the output was saturated. If not, it allows integration to proceed.
Line 3	Sends the (possibly saturated) control output to the plant via the DAQ.
Line 6	Saves the current output for use in the next timestep.
Line 7	Saves the current error value for use in the next timestep.

Table 3: Explanation of anti-windup and state update logic

```

1 double ut_Output = ut_Unsat;
2
3 if (ut_Output > 9.8) { ut_Output = 9.8; }
4 if (ut_Output < -9.8) { ut_Output = -9.8; }

```

Listing 3: Output Saturation Logic

Line Number	Explanation
Line 1	Initializes the output as the controller’s raw output before any limiting.
Line 3	Saturates the output at +9.8 V to stay within actuator limits.
Line 4	Limits the output to -9.8 V to prevent under-driving the actuator.

Table 4: Explanation of output saturation logic

4.4 Clamping Logic

The following describes the function of each block in the output saturation and clamping logic:

- **Output: Clamp**
 - The final output is a Boolean signal that informs the main controller to disable the integrator during periods of actuator saturation.
- **AND Gate**
 - Combines the logic checks from the comparison blocks. It outputs a clamp signal that will stop the integration process if saturation is confirmed.
- **Comparison Blocks (== and ≠)**
 - These blocks check if the **Pre-Sat** and **Pre-Int** signals are equal or not. Any difference between the two signals indicates a saturation event, which will trigger the next steps in the logic.

- **int8 Cast**
 - Converts the signal into 8-bit integers to simplify the comparison process.
- **DeadZone Block**
 - Filters out minor differences that are caused by numerical noise, ensuring only significant saturation events trigger clamping.
- **ZeroGain Block**
 - Passes the signal through a zero gain block, ensuring no feedback while still allowing a comparison of the signal.
- **Inputs: Pre-Sat, Pre-Int**
 - **Pre-Sat**: Represents the unsaturated control signal before clamping.
 - **Pre-Int**: Represents the signal before the integration process occurs.

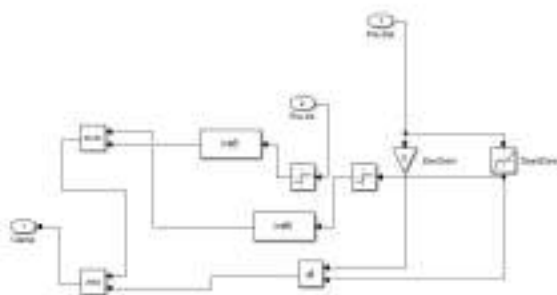


Figure 4: Clamping Logic Diagram

5 Validation

This section presents validation results for the designed control system. Validation was carried out in two stages: first through simulation-based demonstration tests, and second through laboratory experimentation on real hardware. In both settings, the controller was evaluated based on its tracking performance, disturbance rejection capability, and robustness under plant variations.

5.1 Simulink Responses and Analysis

The models depicted in Figures 16, 17 and 19 yielded the following results:

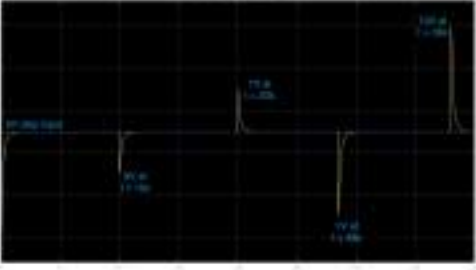
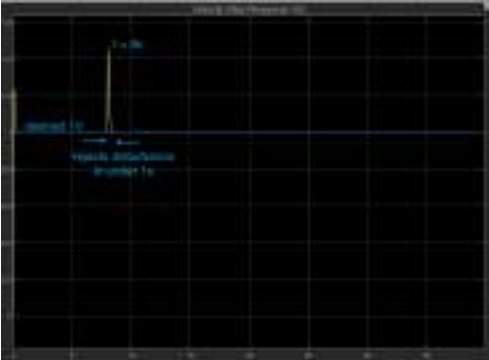

 <p>Figure 5: Position Output Response</p>	<p>The position output plot captures the system’s impressive response to a 5V step input under both nominal conditions and a series of intentional disturbances. From the outset, the controller guides the system smoothly to the target with minimal overshoot and negligible steady-state error. When faced with disturbances—such as a sharp drop at 10s, a sudden spike at 20s, and the rotational effects of mass discs around 28s and 38s—the system reacts quickly and decisively. In each case, the output rapidly returns to the setpoint, highlighting the controller’s excellent tracking ability and robust disturbance rejection.</p>
 <p>Figure 6: Velocity Output Response</p>	<p>The velocity output response showcases a well-tuned and stable system. After a 1V step input at $t=0s$, the system reacts with a quick rise, minimal overshoot, and rapid settling—demonstrating excellent damping characteristics. At around 8s, a disturbance briefly disrupts the output, but the controller responds immediately, bringing the signal back to the reference with minimal deviation. Beyond that point, the response remains clean and consistent, confirming the system’s strong dynamic behaviour and robust control performance.</p>
 <p>Figure 7: QFT Controller Across Plant Variations</p>	<p>This step response vividly demonstrates the QFT controller’s ability to maintain performance across a range of uncertain plant conditions—including the nominal case, increased inertia, and altered brake dynamics. Despite variations in system parameters, all responses successfully converge to the 5V reference. While some transient differences in overshoot and damping are observed, even the most extreme cases settle comfortably within 5 seconds. This consistent behaviour reinforces the controller’s robustness and confirms that the design meets its intended performance criteria.</p>

Table 5: Simulink Response Plots with Corresponding Performance Analysis

5.2 Lab Demonstration Responses and Analysis

Lab-based experiments were conducted to compare the controller’s behaviour in the real-world environment against the simulation results.

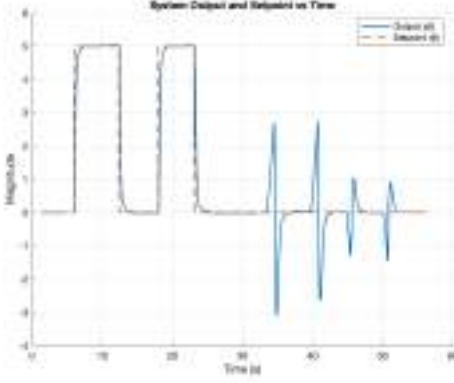
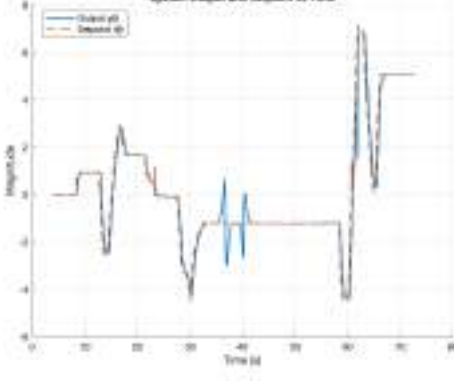
 <p>Figure 8: Tracking and Disturbance Rejection</p>	<p>This simulation showcases the system’s response to a step change in the reference signal—from 0V up to 5V and back down again. The red line indicates the desired setpoint, while the blue curve reflects the plant’s actual output. Throughout the test, four disturbances were introduced at various intervals. Impressively, the controller reacted swiftly each time, restoring the output to the setpoint within 1 second and maintaining precise tracking with virtually no steady-state error. The result highlights the system’s agility and robustness under dynamic conditions.</p>
 <p>Figure 9: Robustness Test under Varying Conditions</p>	<p>This robustness test subjected the system to challenging variations, including a brake-induced friction disturbance and significant changes in attenuation—both high and low. To further test its limits, output disturbances were introduced at $t=35s$ and $t=40s$. Despite these disruptions, the controller consistently maintained accurate reference tracking and swiftly rejected disturbances, confirming its reliable performance across a broad range of plant conditions.</p>

Table 6: Controller Lab Testing Results

5.3 Closing the Triangle of Theory, Simulation and Measurement

A central goal of this project was to achieve coherence between theoretical modelling, simulation-based design, and real-world implementation. This subsection demonstrates that goal being realised: the system’s predicted behaviour in Simulink aligns closely with

the actual performance observed during lab testing. Together, these results confirm that the theoretical control strategy successfully translated into practice.

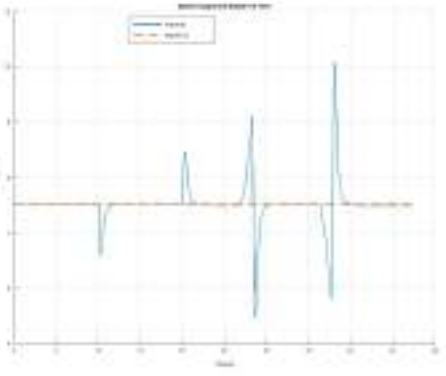
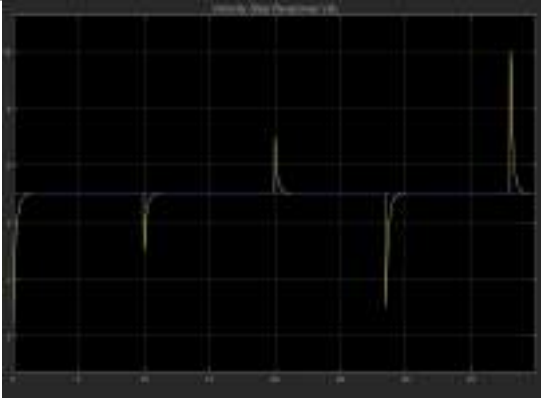
 <p>Figure 10: Log from Physical Servo Motor Test plotted in MATLAB</p>	<p>This figure presents the raw data recorded during the lab test, capturing manually applied disturbances at approximately $t = 28.5\text{s}$ and $t = 38\text{s}$. In both cases, the system responds promptly, returning to the reference with minimal delay.</p>
 <p>Figure 11: Simulink Output for Validation Comparison</p>	<p>The Simulink simulation results closely mirror the lab test responses, exhibiting near-identical transient dynamics and steady-state accuracy. This strong alignment reinforces the consistency between the theoretical design, simulation model, and real-world system performance.</p>

Table 7: Laboratory Validation Summary

These results illustrate that the control system not only performs well in theory and simulation, but also in real-world conditions—successfully “closing the triangle” of design, simulation, and measurement. This convergence affirms the validity of the modelling approach, the soundness of the implementation, and the practical robustness of the overall system.

6 Conclusion

This lab demonstrated the design and implementation of a digital position controller for a servo system using Quantitative Feedback Theory (QFT). By applying frequency-domain

methods like inverse Nichols charts and sensitivity bounds, the controller ensured reliable tracking and disturbance rejection across varying plant conditions.

The controller was implemented digitally with a discrete-time difference equation, addressing challenges such as integrator windup. Anti-windup logic proved effective in maintaining performance near saturation limits.

Both simulation and experimental tests showed good agreement, confirming the accuracy of the model.

This lab provided insight into applying theoretical control strategies to real-time systems, with the final implementation meeting design objectives for robustness, responsiveness, and stability.

7 Appendix

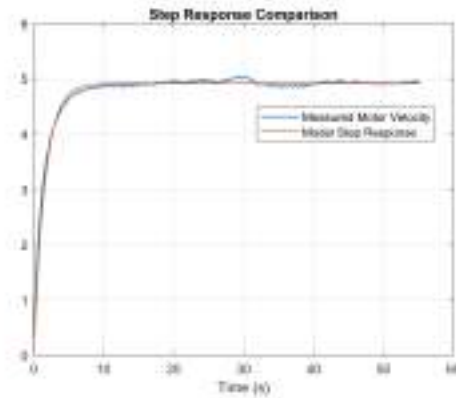


Figure 12: Validation of Identified System Transfer Function via Step Response (raw data response was shifted to the origin for comparison purposes)

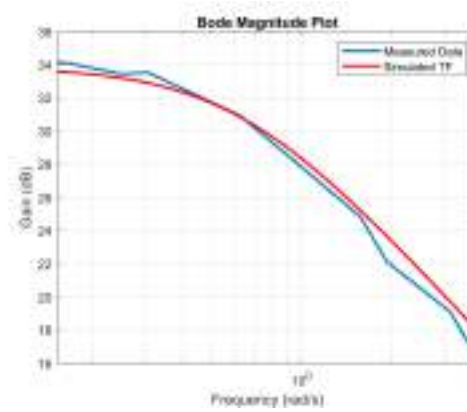


Figure 13: Validation of Identified System Transfer Function via Bode Magnitude Plot Comparison

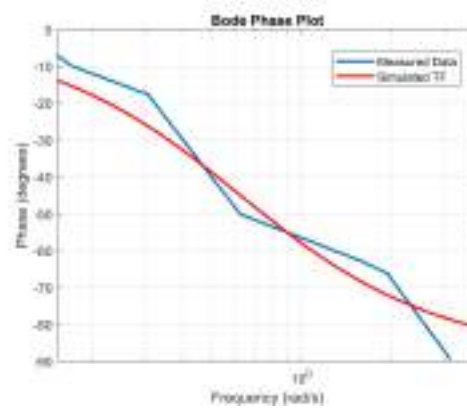


Figure 14: Validation of Identified System Transfer Function via Bode Phase Plot Comparison

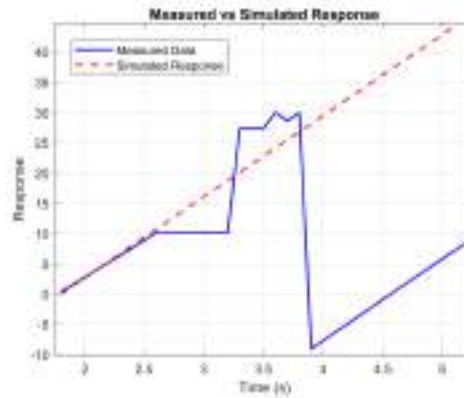


Figure 15: Validation of the Scaling Parameter, a

```

1 // GLOBAL CONTROL VARIABLES
2 double yt_PlantOutput;           // Output of plant for position @ ADC
   INPUT 0
3 double rt_SetPoint = 0;          // Setpoint @ ADC INPUT 1
4 double rt_Gain = 0;              // Gain to change dynamically if needed
5 double ut_PlantInput;           // Output of controller to actuator @ ADC
   INPUT 2
6
7 double Gain = 1;                 // Gain default to one
8
9 // PREVIOUS VALUES
10 double En_1 = 0;
11 double Un_1 = 0;

```

Listing 4: Global Control Variables to set the old values and errors

```

1 if (XValue >= 10 && XValue < 10.03)
2 {
3     yt_PlantOutput += -30; // output disturbance
4 }
5 else if (XValue >= 20 && XValue < 20.03)
6 {
7     yt_PlantOutput += 30; // output disturbance
8 }

```

Listing 5: C# Code for Output Disturbance Injection

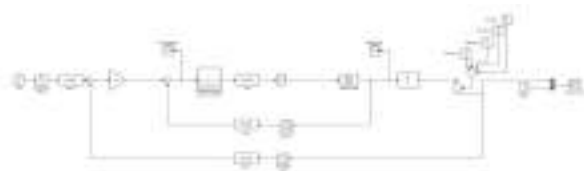


Figure 16: Position Controller Simulink

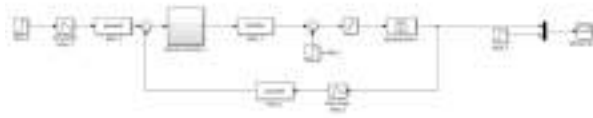


Figure 17: Velocity Controller Simulink Model

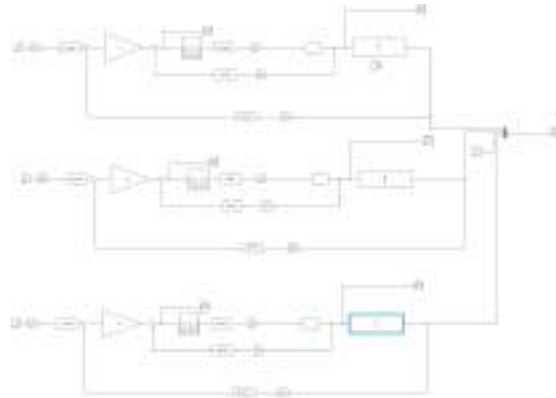


Figure 18: Variations of the Plant Transfer Function

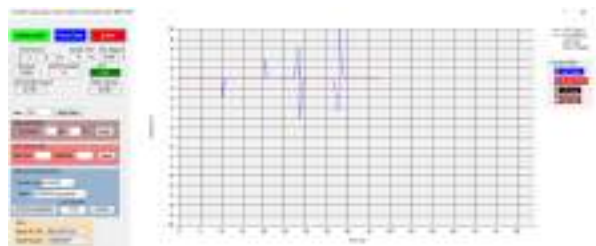


Figure 19: Raw Data from Lab Test for Disturbance Rejection