Control Systems

CS4: Speed Control of a DC Motor

1 Introduction :

The aim of this lab experiment is to design, implement, and test a speed control for a DC motor. Building on the simulation model developed in a previous lab, we will utilize a PWM-based power amplifier to actuate a permanent-magnet DC motor. The motor's speed is measured by a tacho-generator and controlled using two cascaded feedback loops implemented in the Simulink Desktop Real-Time environment. This experiment includes both simulation and real-time control tests, providing a comprehensive evaluation of the designed control scheme.

2 Derivation of plant transfer functions :

As we are using the Cascade Control System (Figure 1) in this context, the "plant" refers to the physical system being controlled, which is the DC motor and its associated components.

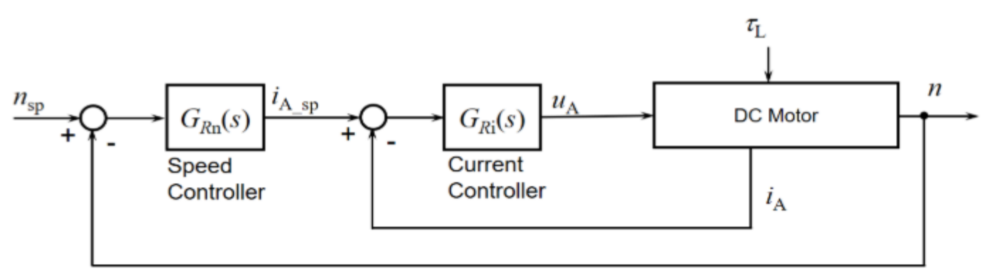


Figure 1: Cascade control

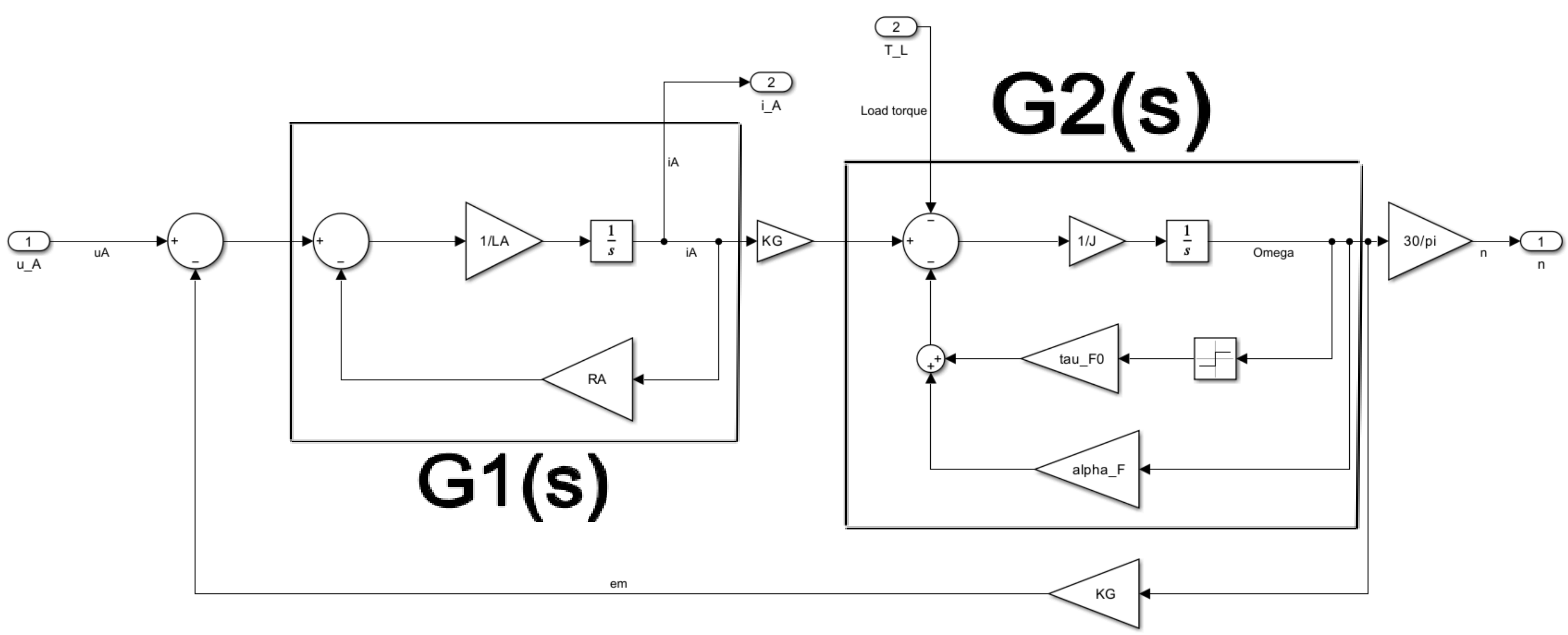


Figure 2: Plant of the system

By neglecting Load Torque we can easily get the transfer functions of the plant ($G_{un}(s)$)

$$G_{un}(s) = \frac{n}{U_A} = \left(\frac{(G_1(s)) .K_G.(G_2(s))}{1+[(G_1(s)).(K_G)^2.(G_2(s))]} \right) \frac{30}{\pi}$$

where,

$$G_1(s) = \left(\frac{\frac{1}{R_A}}{\frac{L_A}{R_A}(s)+1} \right) , \; G_2(s) = \left(\frac{\frac{1}{J}}{\frac{J}{\tau_F}(s)+1} \right) , \; \tau_F = \tau_{F0}(sign(\omega)) + \alpha_F(\omega)$$

3 Details of the controller design :

Step 1: Designing the current controller :-

A PI controller with the following transfer function is used to control the armature current:

$$G_{Ri}(s) = K_{Ri}\left(1 + \frac{1}{T_{Ii}(s)}\right)$$

The controller is designed using the method known as "**modulus optimum method**".This design method optimizes the closed-loop magnitude response. During the design process, the dependency of induced emf e_M on the current itself is neglected (e_M is considered as an unknown disturbance).

This assumption is justified, as the current dynamics are normally much faster than the induced emf, which depends on the speed of the motor.

the following steps to design the current controller:

1. Derive the plant model by neglecting the disturbance e_M :-

$$G_{RL}(s) = \frac{I_A(s)}{U_A(s)} = \left(\frac{\frac{1}{R_A}}{\frac{L_A}{R_A}(s)+1} \right) = \boxed{\left(\frac{0.566}{(0.001981)(s)+1} \right)}$$

R_A=1.7660 Ω and L_A = 3.5 mH (Data from CS2 Lab)

2. Multiply the plant model with the filter transfer function.

$$G_{RLF}(s) = G_{RL}(s).\, G_{Fi}(s) = \boxed{\left(\frac{0.566}{((0.001981)(s)+1)} \right).\left(\frac{1}{0.002(s)+1} \right)}$$

$$G_{Fi}(s) = \left(\frac{1}{T_{Fi}(s)+1} \right),\ \boxed{T_{Fi} = 0.002 \text{ sec}}$$

3. The controller will be implemented on a digital computer with a sample time of T_s=0.001s.

The dead-time caused by the sample and hold process is nearly half of the sample time and can be approximated by first-order lag (PT1). Transfer function of the extended plant can be written as:

$$G_s(s) = G_{RL}(s).\, G_{Fi}(s).\, G_{TD}(s) = \left(\frac{\frac{1}{R_A}}{[\frac{L_A}{R_A}(s)+1][T_{Fi}(s)+1][T_s(0.5)(s)+1]} \right)$$

4. Factors involving two smaller time constants are combined to get an equivalent time constant T_σ

$$G_s(s) = \left(\frac{K_S}{[(T_1)(s)+1][T_\sigma(s)+1]} \right)$$

With $K_S = \frac{1}{R_A} = \boxed{0.566 \frac{1}{\Omega}}$

$$T_1 = \frac{L_A}{R_A} = \boxed{0.001981 \frac{H}{\Omega}}$$

$$T_\sigma = T_{Fi} + 0.5\,(T_s) = 0.002 + 0.5(0.001) = \boxed{0.0025 \text{ sec}}$$

5. The controller parameter T_{Ii} should compensate the larger time constant of $G_s(s)$:

$$T_{Ii} = T_1 = \boxed{0.001981 \frac{H}{\Omega}}$$

6. The controller gain KRi can be calculated according to the following expression:

$$K_{Ri} = \frac{T_{Ii}}{2.T_\sigma.K_S} = \frac{L_A}{2.T_{Fi}+T_S} = \frac{0.001981}{2\ (0.0025)(0.566)} = \boxed{0.700}$$

Step 2: Designing the speed controller :-

The PI controller used to control the motor speed in the outer loop is typically designed using the **symmetrical optimum method**.

controller transfer function:

$$G_{Rn}(s) = K_{Rn}\left(1 + \frac{1}{T_{In}(s)}\right)$$

Plant model required to design this controller can be derived from the block diagram given in Figure 3.

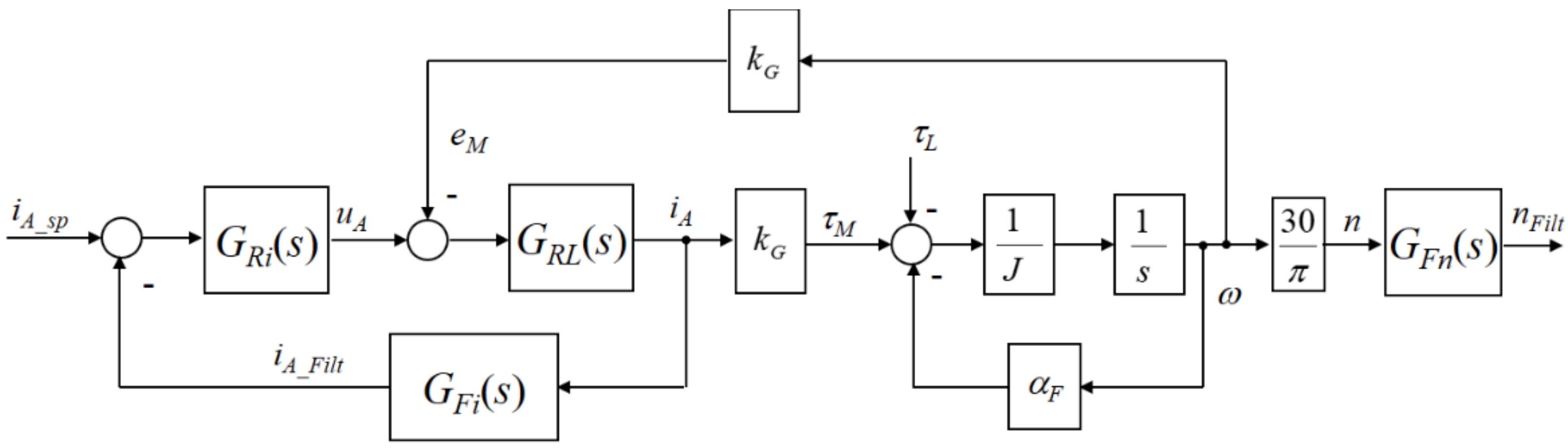


Figure 3: Plant model for the outer loop

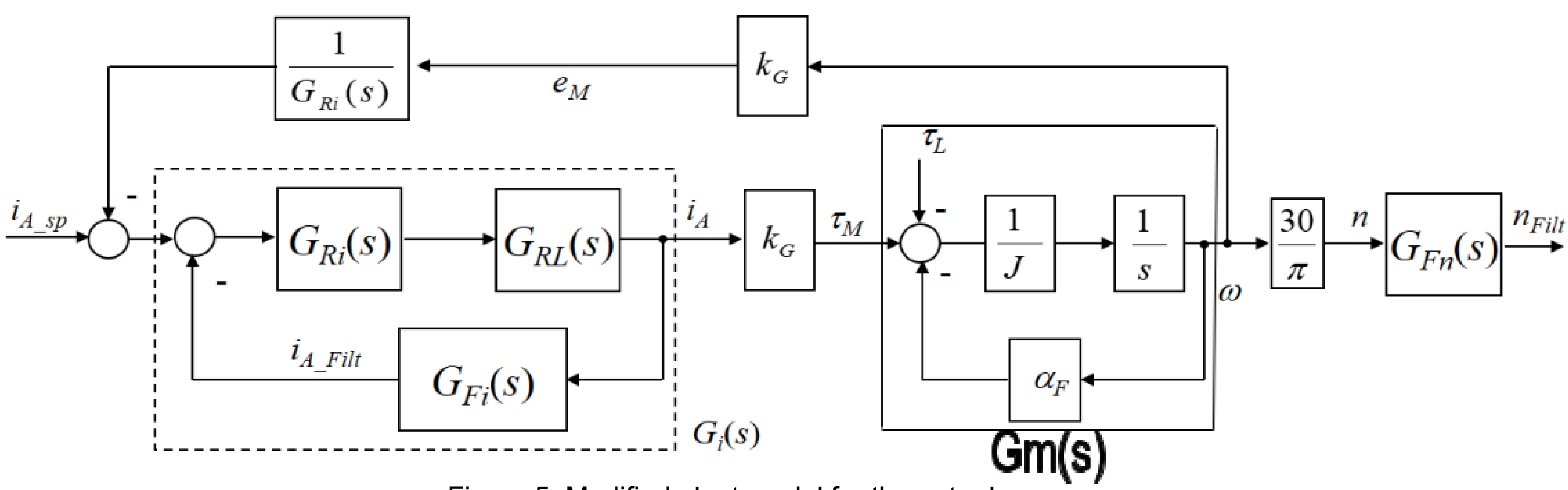


Figure 5: Modified plant model for the outer loop

The speed controller is designed by performing the following steps:

1. Calculate the closed-loop transfer function of the inner loop.

$$G_i(s) = \frac{I_A(s)}{I_{A_{SP}}(s)} = \boxed{\left(\frac{G_{Ri}\,.\,G_{RL}}{1+G_{Ri}\,.\,G_{RL}\,.\,G_{Fi}}\right)} \quad \text{(Inner loop)}$$

2. Calculate the transfer function of the entire model (without speed signal filter):

$$G_m(s) = \left(\frac{\frac{1}{J.(s)}}{1+\frac{\alpha_F}{J.(s)}}\right) = \boxed{\left(\frac{\frac{1}{\alpha_F}}{1+\frac{J}{\alpha_F}(s)}\right)}$$

$$G_n(s) = \frac{N(s)}{I_{A_{SP}}(s)} = \boxed{\left(\frac{G_i\,.\,K_G\,.\,G_m}{1+\left(\frac{G_i\,.(K_G)^2\,.\,G_m}{G_{Ri}}\right)}\right)}$$

3. Also consider the speed signal filter

$$G_{Fn}(s) = \left(\frac{1}{T_{Fn}(s)+1}\right) = \boxed{T_{Fn} = 0.02 \text{ sec}}$$

$$G_{Sn}(s) = G_n(s).G_{Fn}(s) = \boxed{\left(\frac{G_i \,.\, K_G \,.\, G_m}{1 + \left(\frac{G_i \,.(K_G)^2 \,.\, G_m}{G_{Ri}}\right)}\right) . \left(\frac{1}{0.02(s)+1}\right)}$$

4. Determine the values of controller parameters K_{Rn} and T_{In} using the symmetrical optimum method with a phase margin of ϕ_m = 37°. This small value of the phase margin is helpful for disturbance rejection. In case of the tracking control showing a large overshoot, the set-point can be filtered using a low-pass filter like PT1.

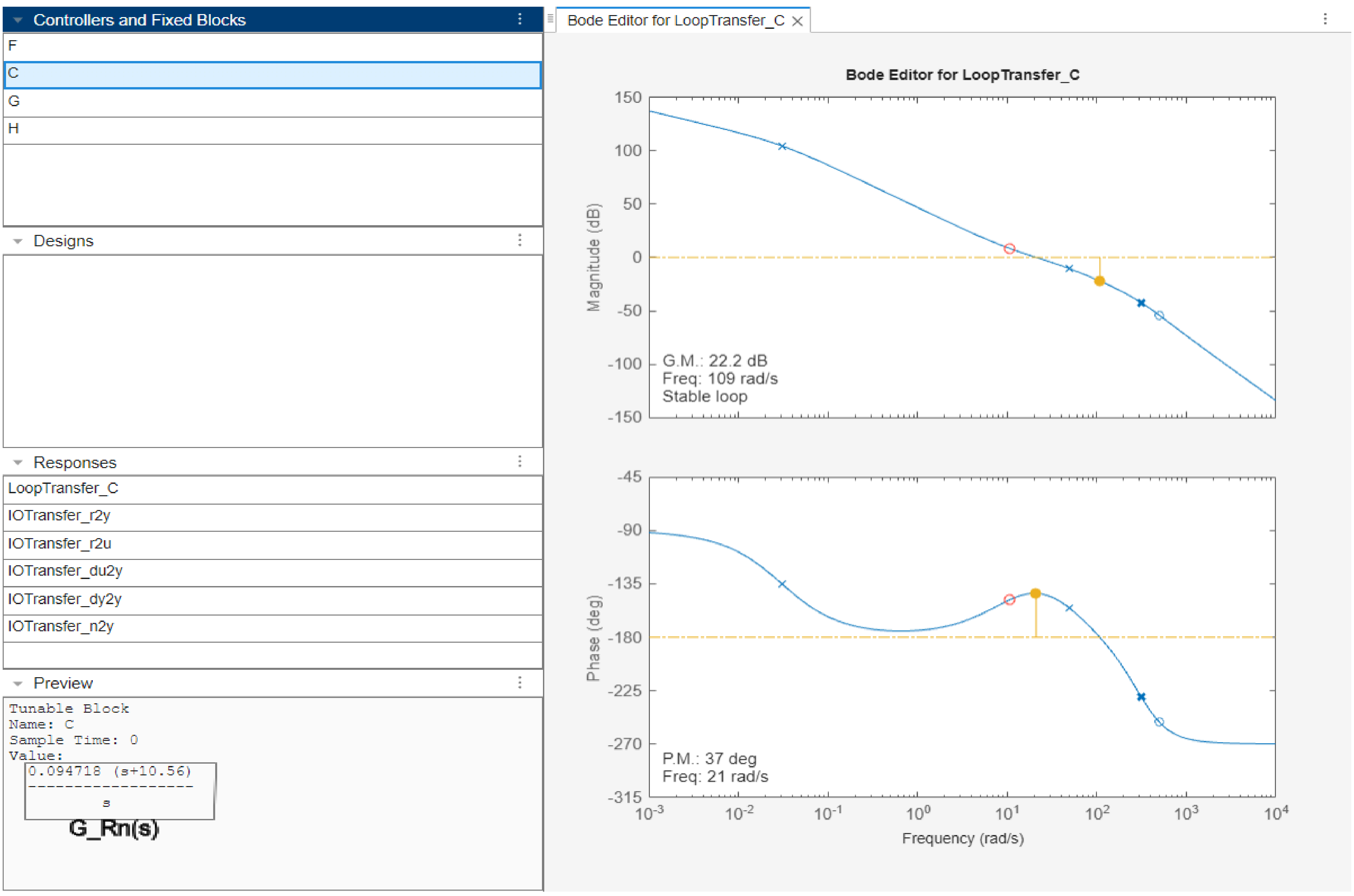


Figure 6: A screenshot of the sisotool

As, $G_{Rn}(s) = K_{Rn}\left(1 + \frac{1}{T_{In}(s)}\right)$

$$\boxed{\therefore\ K_{Rn} = 0.09471 \text{ and } T_{In} = 0.09469}$$

$$\boxed{\begin{array}{l} \therefore\ K_{awi} = K_{Ri} = 0.700 \\ K_{awn} = 50\ .\ K_{Rn} = 4.7355 \end{array}}$$

4. Graphical representation of simulation results

5. Graphical representation of real-time control of the motor

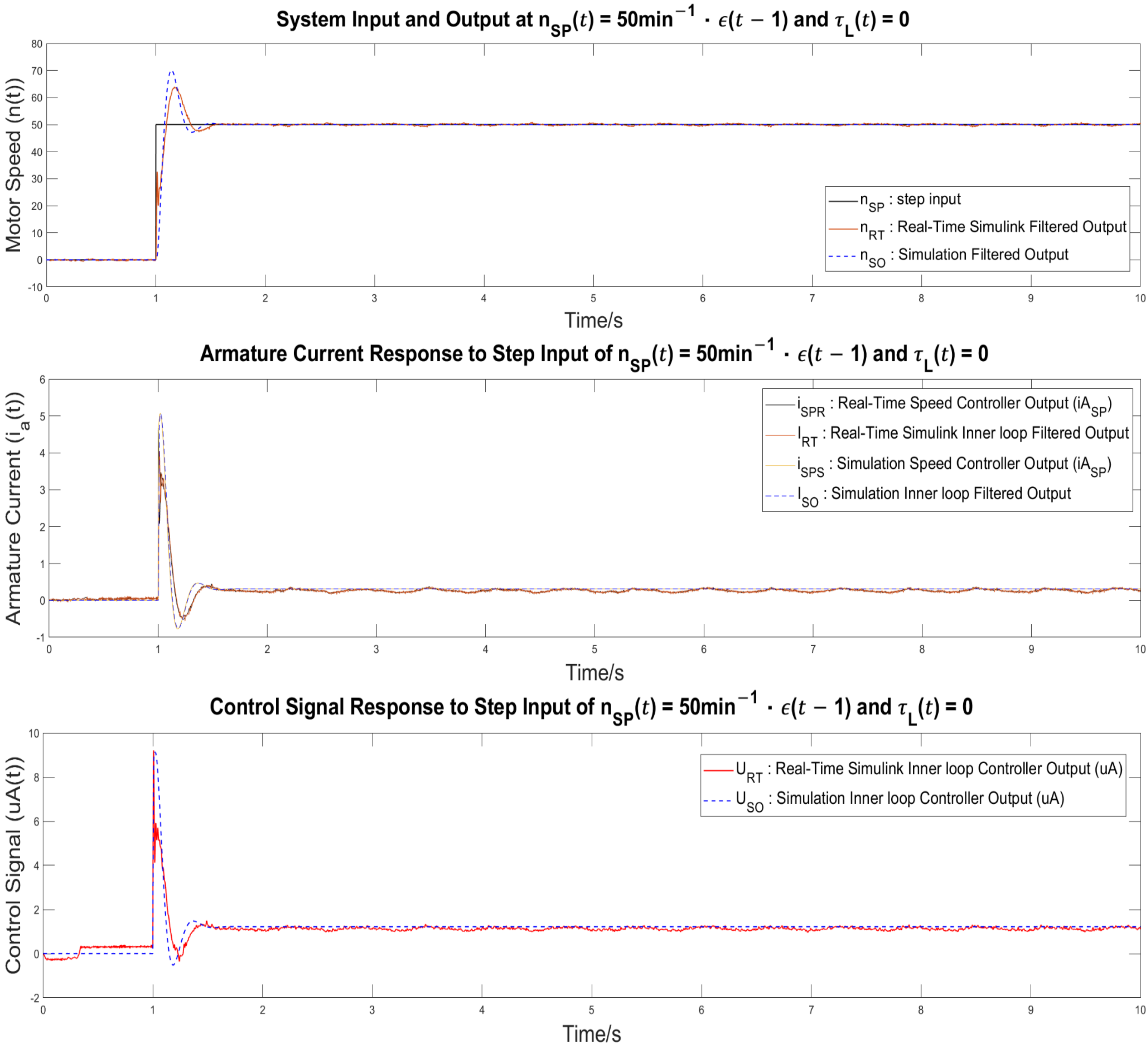
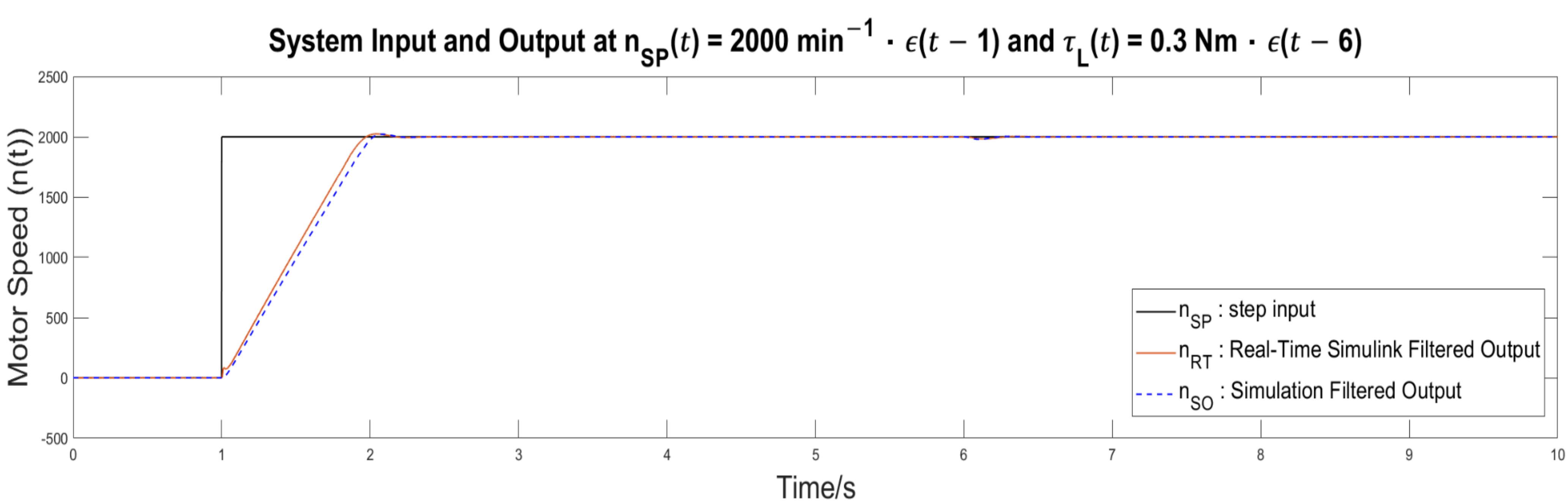
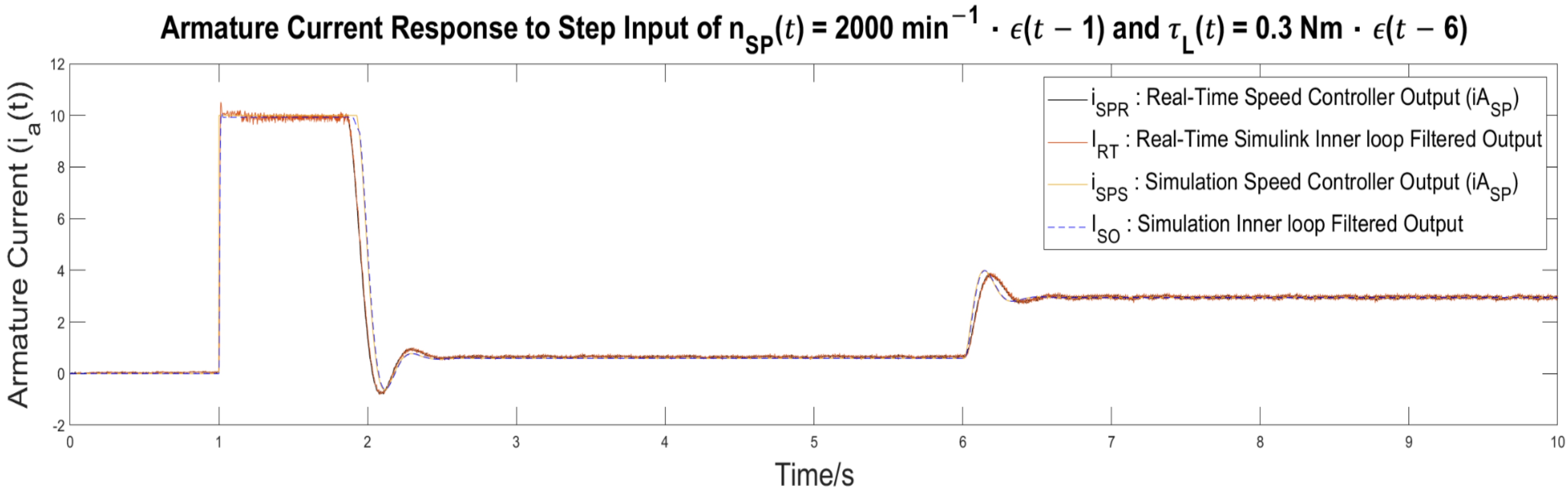


Figure 7 : Graphical representation of simulation results and real-time control of the motor ($n_{SP}(t) = 50min^{-1} \cdot \epsilon(t-1)$ and $\tau_L(t) = 0$)





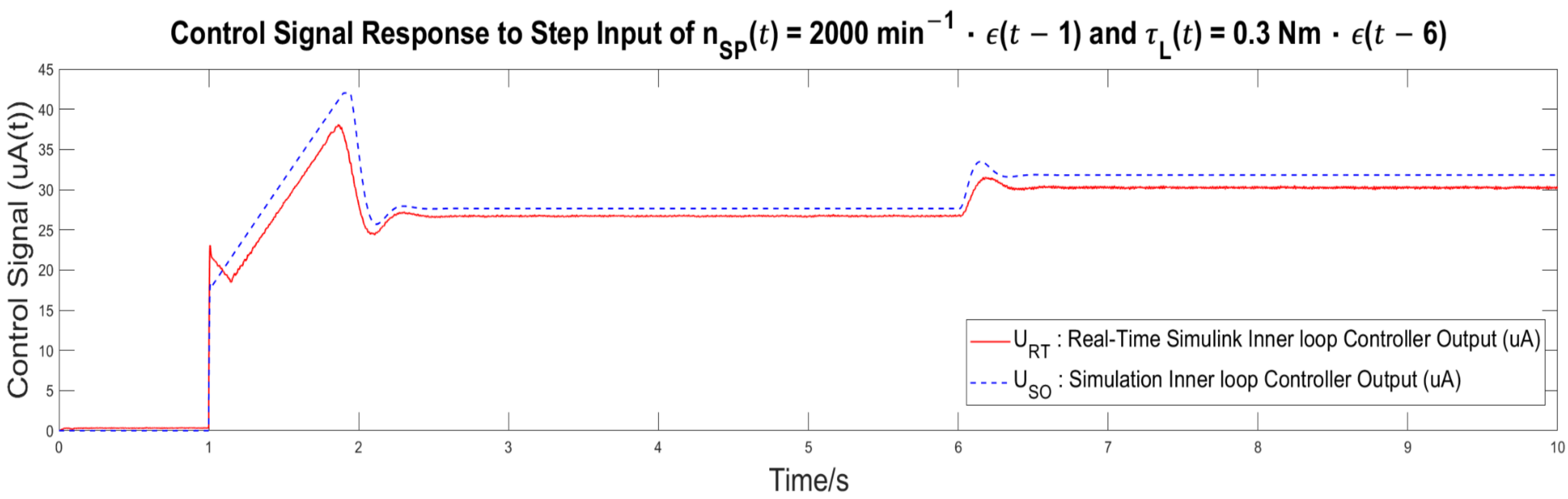


Figure 8 : Graphical representation of simulation results and real-time control of the motor
$(n_{SP}(t) = 2000 min^{-1} \cdot \epsilon(t - 1)$ and $\tau_L(t) = 0.3$ Nm $\cdot\ \epsilon(t - 6))$

6 Conclusion :

In this experiment, we investigated the performance of a DC motor speed control system under various conditions and compared the simulation results with real-time implementation. The following key observations and conclusions were drawn:

1. **Setpoint Response and Overshoot**:
    - **Smaller Setpoint**: The system exhibits noticeable overshoot when a smaller setpoint is applied (Figure 7). This is attributed to the system's aggressive response to quickly reach the desired speed. The initial overshoot indicates that the controller parameters might need further tuning for lower setpoints to minimize overshoot while maintaining a fast response time.
    - **Larger Setpoint**: When a larger setpoint is set, the system transitions smoothly to the desired speed with almost not observable overshoot (Figure 8). This behavior suggests that the controller is more effective at managing larger setpoint changes, potentially due to the damping effects of the load and the inherent characteristics of the motor and control system.

2. **Disturbance Rejection**:

   Around the 6th second mark in both the simulation and real-time results(Figure 8), a small deflection is observed when the disturbance load is applied. Despite this, the system quickly returns to the setpoint, demonstrating effective disturbance rejection capabilities of the control system. The minimal deflection indicates that the control system can handle sudden changes in load efficiently, maintaining stable operation.

3. **Steady-State Accuracy and Noise:**

   In theory, a PI controller should achieve 100% accuracy in steady-state conditions. However, real-time observations indicate the presence of noise around the setpoint, particularly when the setpoint is constant at 10A. This noise could be due to several factors:

   - Integral Action (Ki): The integral component of the PI controller continuously accumulates error over time. If there is a sudden change in the setpoint, the accumulated error may cause oscillations or noise as the controller tries to correct it.
   - Digital-to-Analog and Analog-to-Digital Converters: Imperfections or quantization errors in these converters can introduce noise into the control signal, affecting the overall stability and accuracy.
4. **Comparison of Simulation and Real-Time Results:**
   - No Load, Low Motor Speed: The real-time system performed better than the simulation, with smoother transitions and less overshoot. This suggests that the simulation model might overestimate the system's instability under no load conditions, highlighting the need for refining the model.
   - With Load, High Motor Speed: Both simulation and real-time results were closely aligned, demonstrating smooth transitions, minimal overshoot, and consistent control behavior. This validates the accuracy of the simulation model and the effectiveness of the control strategy under load conditions.