CARLETON UNIVERSITY
Department of Systems and Computer Engineering

SYSC 3600
Lab #2: Servo System Simulation

Before coming to Lab #2:

- Read and understand the below pre-lab for Lab #2.
- Read the "SIMULINK/MATLAB Extras" in Appendix A.
- Familiarize yourself with the SIMULINK models given in Appendix B.

Instructions:
Read through and complete the entire lab. Your report should address the points made in the **Report** suggestion boxes and demonstrate your understanding of the material.

---

1 Purpose

The purpose of this lab is:

(a) To study a servo system which is an important dynamic system occurring frequently in industrial applications.

(b) To gain further experience in digital simulation using SIMULINK.

2 Pre-lab

2.1 Servo System

A common type of control system is a *servomechanism* in which a mechanical output variable, such as the position or angular rotation of an element, is required to follow a reference input. An example of a servomechanism is given in Fig. 1 where an input dial is used to control the angular position of an output shaft.[1] An example use of this servo system could be the manual azimuth control of a telescope or a directional antenna (*i.e.*, rotate the telescope on its base according to the position of the input dial.).

The way that this servomechanism works is as follows. The input potentiometer is used as a variable voltage divider, where turning a dial causes the *wiper arm* to make contact at different positions along the potentiometer's resistive element. Depending on the position of the wiper arm, the voltage division across the resistive element generates the voltage signal $e_1(t)$, which has a range in $[-V, +V]$.

The output potentiometer works in the same fashion to generate $e_2(t)$. The only difference is that the position of the wiper arm on the output potentiometer is controlled by rotation of the shaft by the motor. The input might typically be some user controlled dial, while the output of the motor (and hence what is attached to the output potentiometer) might be a very heavy object that needs turning.

[1] Most armature control systems would also include an inductor in series with the motor, but this is left out in this example to simplify the model.

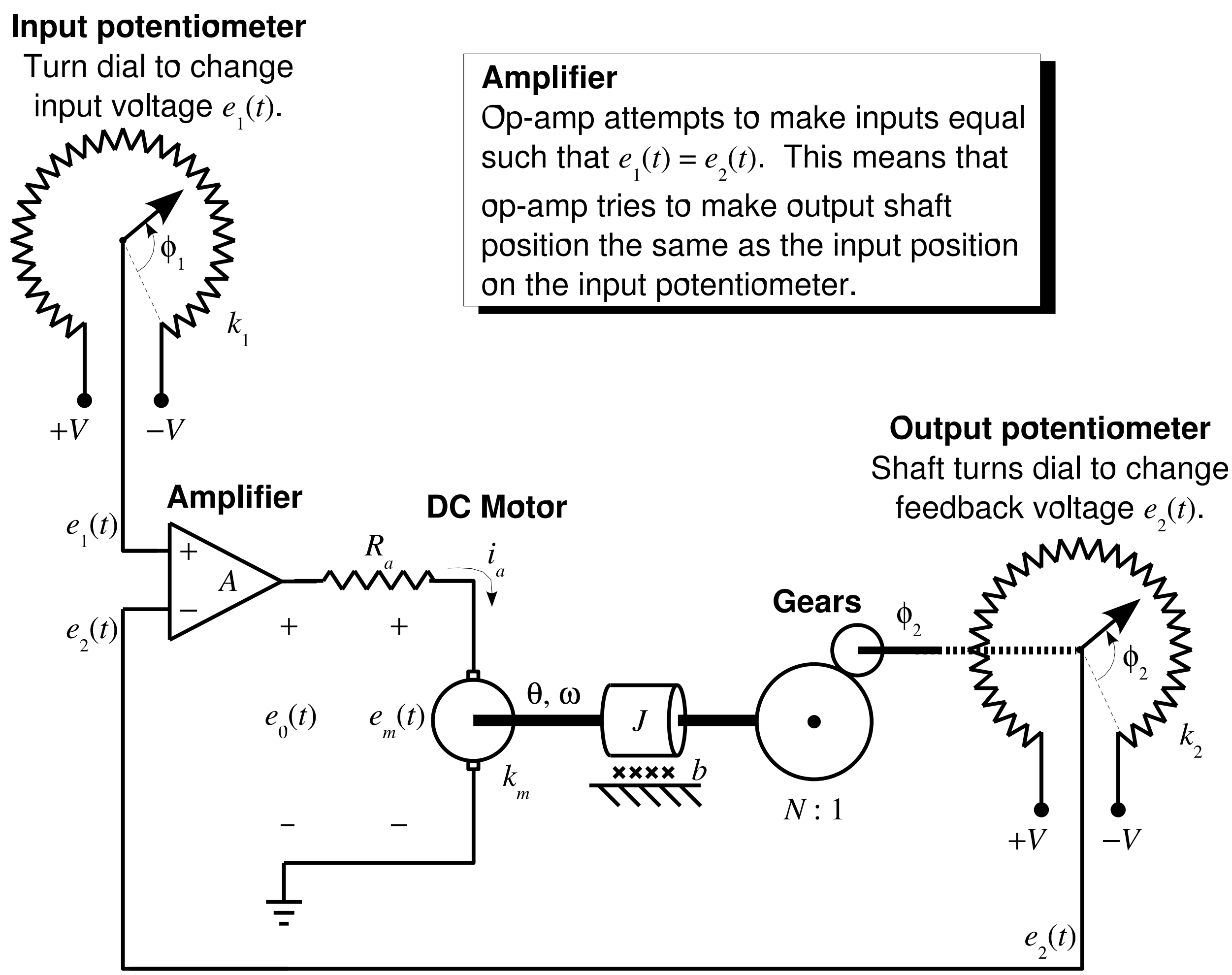

Figure 1: Servo system[2] where turning the input dial ϕ_1 makes the motor turn the output shaft ϕ_2 to the same angular position.

The two signals $e_1(t)$ and $e_2(t)$ act as indicators of the angle $\phi_1(t)$ of the input and the angle $\phi_2(t)$ of the output. These two signals are connected to an amplifier that computes the difference between them as follows

$$e_0(t) = A\,[e_1(t) - e_2(t)]\,. \tag{1}$$

Notice that if $e_1(t) = e_2(t)$, then $e_0(t) = 0$, in which case the driving voltage across the motor is zero volts. Hence, if the input and the output are at the same position, then the motor leaves the output alone. If $e_1(t) > e_2(t)$, then $e_0(t)$ is positive causing the motor to turn the output to the positive direction making $e_2(t)$ gradually larger. Eventually, $e_2(t)$ should reach $e_1(t)$ which will cause the motor to stop. Similarly, if $e_1(t) < e_2(t)$, then $e_0(t)$ is negative causing the motor to turn in the opposite direction.

You should notice that all the amplifier constant A will affect is how much to amplify the difference between $e_1(t)$ and $e_2(t)$, effectively making the motor go faster for larger A or slower for smaller A.

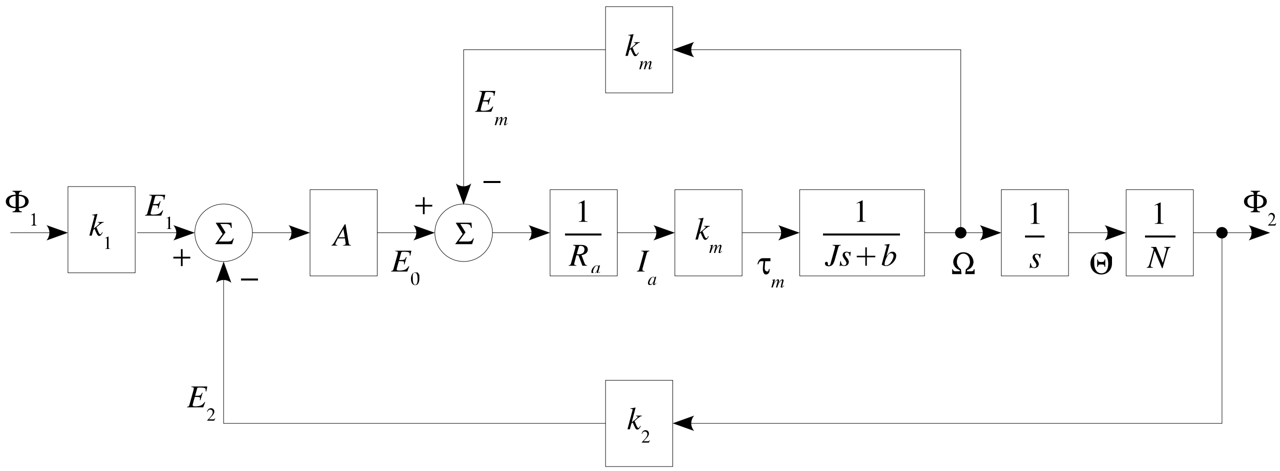

Figure 2: Block diagram of the servomechanism illustrated in Fig. 1.

The equations (and their Laplace transforms) governing the system given in Fig. 1 are as follows:

$$e_1 = k_1\phi_1 \quad \stackrel{\mathscr{L}}{\Longleftrightarrow} \quad E_1(s) = k_1\Phi_1(s) \tag{2}$$

$$e_o = A(e_1 - e_2) \quad \stackrel{\mathscr{L}}{\Longleftrightarrow} \quad E_o(s) = A[E_1(s) - E_2(s)] \tag{3}$$

$$i_a = \frac{e_o - e_m}{R_a} \quad \stackrel{\mathscr{L}}{\Longleftrightarrow} \quad I_a(s) = \frac{E_o(s) - E_m(s)}{R_a} \tag{4}$$

$$\tau_m = k_m i_a \quad \stackrel{\mathscr{L}}{\Longleftrightarrow} \quad T_m(s) = k_m I_a(s) \tag{5}$$

$$\tau_m = J\dot{\omega} + b\omega \quad \stackrel{\mathscr{L}}{\Longleftrightarrow} \quad T_m(s) = (Js + b)\Omega(s) \tag{6}$$

$$e_m = k_m\omega \quad \stackrel{\mathscr{L}}{\Longleftrightarrow} \quad E_m(s) = k_m\Omega(s) \tag{7}$$

$$\omega = \dot{\theta} \quad \stackrel{\mathscr{L}}{\Longleftrightarrow} \quad \Omega(s) = s\Theta(s) \tag{8}$$

$$\phi_2 = \frac{\theta}{N} \quad \stackrel{\mathscr{L}}{\Longleftrightarrow} \quad \Phi_2(s) = \frac{\Theta(s)}{N} \tag{9}$$

$$e_2 = k_2\phi_2 \quad \stackrel{\mathscr{L}}{\Longleftrightarrow} \quad E_2(s) = k_2\Phi_2(s) \tag{10}$$

Verify that you know how to determine Eq. 2 to Eq. 10 that govern the system illustrated in Fig. 1. Also, verify that you know how to find the Laplace transform of each governing equation.

2.2 Block Diagram of the Servo System

Given the governing equations in Eq. 2 to Eq. 10 of the servomechanism, it is reasonably straightforward to develop a block diagram. The block diagram in Fig. 2 directly realizes each of the governing equations, and hence realizes the overall servomechanism model. A SIMULINK subsystem realization of this block diagram is given in Fig. 9 in Appendix B.

[2]Simplified version of Fig. 8–29, Ogata, *System Dynamics, 3/e*, or Fig. 10-29, Ogata, *System Dynamics, 4/e*.

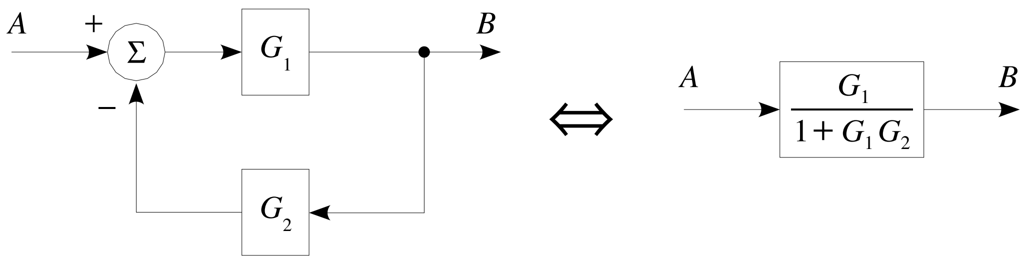

Figure 3: Block reduction of a negative feedback loop.

Verify that you can derive the block diagram yourself from the governing equations of the servomechanism. Start by drawing a block diagram for each of the governing equations Eq. 2 to Eq. 10, while making appropriate connections between block diagram segments.

2.3 Block Diagram Reduction

While we could simulate the block diagram given in Fig. 2 directly in SIMULINK, it is sometimes advantageous to manipulate the block diagram through block reduction. Block reduction can be used to combine blocks in series or in parallel to make the overall block diagram easier to implement.

2.3.1 Negative feedback

One important block reduction relationship is given in Fig. 3. In this block reduction, a negative feedback loop is reduced to a single block. Notice from the left side of Fig. 3 that

$$B(s) = G_1(s)\,[A(s) - B(s)G_2(s)] \tag{11}$$

Solving for the transfer function $H(s) = \frac{B(s)}{A(s)}$ of this negative feedback loop gives

$$H(s) = \frac{B(s)}{A(s)} = \frac{G_1(s)}{1 + G_1(s)G_2(s)}. \tag{12}$$

2.3.2 Block diagram reduction of motor/axle

Using the notation in Sec. 2.3.1, notice that the inner negative feedback loop in Fig. 2 can be expressed with

$$G_1(s) = \frac{k_m}{R_a(Js + b)} \tag{13}$$

$$G_2(s) = k_m \tag{14}$$

Putting $G_1(s)$ and $G_2(s)$ into the negative feedback loop block reduction equation given by Eq. 12 gives

$$H_{inner}(s) = \frac{\dfrac{k_m}{R_a(Js+b)}}{1 + k_m \dfrac{k_m}{R_a(Js+b)}} = \frac{k_m}{R_a Js + R_a b + k_m^2} \tag{15}$$

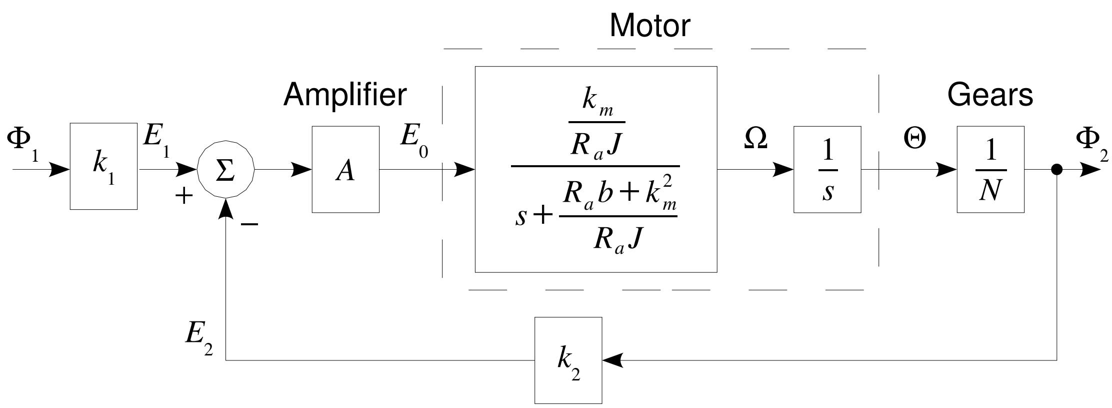

Figure 4: Block reduction of simulation diagram to amplifier, motor, and gears.

Sometimes it is preferred that transfer functions are given with the coefficient associated with s^N as 1, where N is the order of the system. Putting the inner feedback loop transfer function in this form gives

$$H_{inner}(s) = \frac{\dfrac{k_m}{R_a J}}{s + \dfrac{R_a b + k_m^2}{R_a J}}. \tag{16}$$

Note that for SIMULINK/MATLAB that either form can be used since Eq. 38 in Appendix A can use either case.

With this block reduction of the inner negative feedback loop, we can redraw the simulation diagram as given in Fig. 4. This block diagram shows how the servomechanism can be represented in terms of the amplifier A, the control portion of the DC motor, and the gearing to the output shaft with $1/N$. The gains k_1 and k_2 are simple gains on the input and the feedback output from the potentiometers. Sometimes reducing blocks into physically identifiable units is helpful since it then allows for the units to be designed separately, or interchanged with other similar units.

Verify that you can perform the block diagram reduction to reduce Fig. 2 to Fig. 4.

2.3.3 Overall transfer function through block reduction

If we wish to determine the transfer function for the overall system, we could continue the block reduction on Fig. 4. To simplify the notation, lets collect some of the constants in Fig. 4 by defining the following constants.

$$B = \frac{k_m}{R_a J} \tag{17}$$

$$C = \frac{R_a b + k_m^2}{R_a J} \tag{18}$$

With this simplification in notation, we can block reduce the negative feedback loop in Fig. 4 using Eq. 12 with

$$G_1(s) = \frac{AB/N}{s(s+C)} \tag{19}$$

$$G_2(s) = k_2. \tag{20}$$

The transfer function of the servomechanism is therefore

$$\begin{aligned}
H(s) &= k_1 \frac{G_1(s)}{1 + G_1(s)G_2(s)} \\
&= k_1 \frac{\dfrac{AB/N}{s(s+C)}}{1 + \dfrac{AB/N}{s(s+C)} k_2} \\
&= k_1 \frac{AB/N}{s(s+C) + k_2 AB/N} \\
&= \frac{k_1 AB/N}{s^2 + Cs + k_2 AB/N}. \qquad (21)
\end{aligned}$$

Expanding out Eq. 21 using B and C gives

$$H(s) = \frac{\dfrac{Ak_1 k_m}{R_a J N}}{s^2 + \dfrac{R_a b + k_m^2}{R_a J} s + \dfrac{Ak_2 k_m}{R_a J N}}. \qquad (22)$$

Perform the block reduction from scratch yourself. You should be able to do it in about 5–8 minutes after some practice. It is not uncommon for students to spend 2–3 hours (yes, that is hours) on this simple block reduction, so make sure you practice and are able to do it in only a few minutes.

2.4 Review of Parameters for a 2^{nd}-Order System

It should be clear that the servomechanism described by Eq. 22 is a 2^{nd}-order system and that this transfer function fits the form

$$H(s) = \frac{k\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad (23)$$

where k is the static gain, ζ is the damping ratio (factor), and ω_n is the undamped natural frequency (in rads/s).

For Eq. 22, give equations to calculate the undamped natural frequency ω_n, the static gain k, and the damping ratio ζ. You will need these equations for the lab.

$$\omega_n = \qquad (24)$$

$$k = \qquad (25)$$

$$\zeta = \qquad (26)$$

The static gain k gives the steady-state gain of the system. The damping ratio ζ indicates the behaviour of the step response with the following cases:

- $\zeta > 1$: Overdamped, characterized by a slow step response with a large time constant but no overshoot/oscillations.
- $\zeta = 1$: Critically damped, characterized by the fastest rise time in the step response for the current choice of ω_n when no overshoot/oscillations occur.
- $0 < \zeta < 1$: Underdamped, characterized by a step response that is "too fast" such that overshoot occurs causing response oscillations.

If $\zeta = 0$, then the step response is a pure sinuoid with a frequency of ω_n (hence why ω_n is known as the *undamped* natural frequency). If ζ is non-zero, then the undamped natural frequency ω_n does not have any physical meaning, but we can compute the damped natural frequency ω_d as follows

$$\omega_d = \omega_n \sqrt{1 - \zeta^2} \text{ rads/s.} \tag{27}$$

The advantage of ω_d is that if it is real then it gives the frequency of the damped oscillation in the step response. Notice that ω_d is complex if $\zeta \geq 1$, which indicates no oscillations in the step response.

2.5 Adjusting the Damping Ratio with a Rate Feedback Loop

From the equations you developed in Eq. 24 to Eq. 26, you can see how the parameters of the system affect the damping ratio ζ, the static gain k, and the undamped natural frequency ω_n. Often we want to adjust ζ and ω_n independently of each other. Unfortunately, many of the same parameters control both ζ and ω_n so it is difficult to design a servo with specific values for ζ and ω_n, particularly when parameters such as k_m, J, b, and N may not be under your control. Typically only R_a, A, k_1, and k_2 will be under your control in your design, but these are linked tightly with both ω_n and ζ.

One common approach to help solve this problem is to introduce a *rate feedback loop* (or *velocity feedback* for this system) as shown in Fig. 5. The rate feedback loop takes the derivative of the angular position using the $\boxed{s}$ block and then multiplies by some constant K to control the level of rate feedback. The rate feedback loop allows the servo to use not only the current position $\phi_2(t)$ of the output shaft but also the velocity $\omega(t)$ of the shaft in the feedback.

In an implementation, the $\boxed{s}$ block should be avoided since differentiating a signal will accentuate any noise. We can instead take the velocity $\omega(t)$ of the shaft before the integration block as shown in Fig. 6 to avoid the differentiation. For the actual implementation, a tachometer would be attached to the motor to measure the velocity of the axle's rotation. A tachometer is simply a small d.c. generator that outputs a voltage proportional to the velocity of the axle. This voltage can be expressed as $e_r(t) = k_r\omega(t)$ volts, where k_r is just the constant of proportionality for that particular tachometer. This output is fed back to the amplifier as shown in Fig. 6, so that $e_0(t) = A(e_1(t) - e_2(t) - k_r\omega(t))$. In this configuration the differentiation of the axle's position is avoided since the velocity of the rotation is measured directly.

Perform block reduction on Fig. 6 to obtain a transfer function $H_{rate}(s)$ and then determine equations for ω_n, k, and ζ. You should obtain the following transfer function.

$$H_{rate}(s) = \frac{k_1AB/N}{s^2 + (C + ABk_r)s + k_2AB/N} \tag{28}$$

Notice that k_r can now be used to adjust ζ without affecting ω_n.

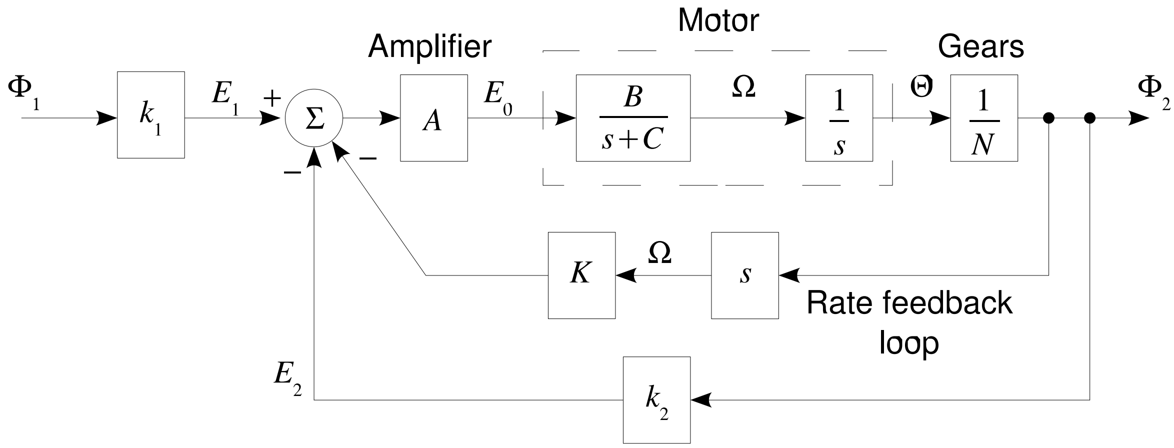

Figure 5: Servo's simulation diagram with a rate feedback loop.

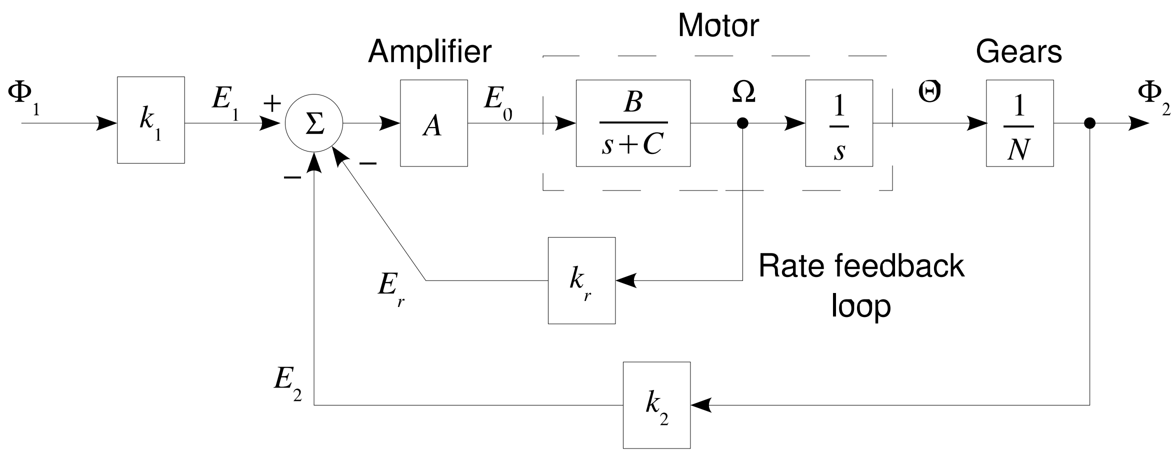

Figure 6: Servo's simulation diagram with rate feedback loop implemented with a tachometer attached to the motor.

For Eq. 28, give equations to calculate the undamped natural frequency ω_n, the static gain k, and the damping ratio ζ. You will need these equations for the lab.

$$\omega_n = \tag{29}$$

$$k = \tag{30}$$

$$\zeta = \tag{31}$$

3 Lab Experiment

3.1 Values to use for the Servo Components

For the servo experiments in the following sections, use the component/parameter values given below.

$$\begin{aligned} k_m &= 1.5275 \text{ kg} \cdot \text{m}^2/\text{sec}^2/\text{A} \quad \text{(Nm/A)} \\ J &= 100 \text{ kg} \cdot \text{m}^2 \\ b &= 100 \text{ kg} \cdot \text{m}^2/\text{sec} \\ R_a &= 1\ \Omega \\ N &= 12 \\ k_1 &= 12 \text{ V/rad} \\ k_2 &= k_1 \end{aligned} \tag{32}$$

You may wish to use a MATLAB script to load these values as described in Appendix A.3.

3.2 Experiments with the Servo using Position Feedback

3.2.1 Verifying the transfer function with the block diagram implementation

Let's start with implementing the block diagram in Fig. 2 as the subsystem shown in Fig. 9. To help verify that the implementation is correct, as well as see if we have the correct transfer function $H(s)$ in Sec. 2.3.3, let's implement both of these as given in Fig. 10 and see if they give the same (or at least very close) results. In Fig. 10, the difference between the step responses of the block diagram and the transfer function are simulated. Ideally, the difference should be zero, but numerical rounding during the computations may give some small non-zero values such as 1.0×10^{-15}.

Note: You can use the component values given in Sec. 3.1 and set A to some non-zero value such as 10.

3.2.2 Setting up the experimental cases

Using the values for the servo given in Sec. 3.1, fill in the following table given the values for the amplifier A in the left-hand column. You should use the equations from Sec. 2.4 to compute these values. For the "Behaviour" column, indicate one of (a) overdamped, (b) critically damped, or (c) underdamped, to describe the system's step response.

Case	A	ζ	ω_n	ω_d	Behaviour
1	4				
2	17				
3	35				
4	300				

You might find it convenient to write a MATLAB script file that will compute these values. The following partial MATLAB script will get you started.

```
% Define variables for Simulink model of the servomechanism.
km=1.5275;
```

```
J=100;       % Moment of inertia of axle
b=100;       % Damping friction on axle
Ra=1;
N=12;        % Gear ratio
k1=12;
k2=k1;       % Use same gain for both potentiometers
B=km/(Ra*J);
C=(Ra*b+km*km)/(Ra*J);

% Set amplifier value
A=4;         % Change this value for different A

wn = sqrt(k2*A*B/N);
damping =         % Fill in this line
wd =              % Fill in this line
k =               % Fill in this line
```

Report: Your report should include the values you calculated in the table in this section.

3.2.3 Simulating the step response of the servo with position feedback

In this section, you will simulate the step response of the servo. Before doing the simulation, compute the final value expected given the input $\phi_1(t) = u(t)$ where $u(t)$ is defined as the unit step function. Fill in the following table with the final value of the step response for each value of A. Recall that the final value of the step response is given as

$$\text{f.v.} = \lim_{t\to\infty} \phi_2(t) \tag{33}$$

with the input a unit step function in this case. Since we don't readily have an equation for $\phi_2(t)$, we can use the final value theorem of the Laplace transform and calculate

$$\text{f.v.} = \lim_{t\to\infty} \phi_2(t) = \lim_{s\to 0} s\Phi_2(s) = \lim_{s\to 0} s\left[H(s)\Phi_1(s)\right]. \tag{34}$$

Case	A	Final Value
1	4	
2	17	
3	35	
4	300	

Now that you have computed some of the characteristics for the servo for four values of A, let's simulate the servo to find its step response. We can then compare the simulation results to what is expected according to the parameters ζ, ω_n, ω_d, and the final value. Simulate and plot the step response for each value of A and make observations for each result compared to what is expected from the two previous tables. For the simulation, you can simulate the servo subsystem as shown in Fig. 11 or using a transfer function block instead to realize $H(s)$.

Some points you should look at in your observations are:

- Does the final value agree with what you calculated with the final value theorem?
- Does the behaviour of the step response agree with what was expected according to the damping ratio ζ?
- In those cases where there are oscillations, is the frequency of the oscillations the same as what you calculated for ω_d? To make it easy to verify, try converting ω_d into a time between peaks in a sinusoid as follows

$$\text{time between peaks} = \frac{2\pi}{\omega_d} \text{ seconds.} \tag{35}$$

  Then you can simply zoom in on the plots and look at the time difference from one peak to the next.

> **Report:** Your report should include the step response for each of the four cases of A, along with discussions on the step responses. Also, your report should address the questions given and you should verify that that the step response obtained matches the calculated values for final value, damping ratio ζ, and damped natural frequency ω_d.

3.3 Experiments on the Servo with Position and Rate Feedback

3.3.1 Verifying the transfer function with the block diagram implementation

Similar to what was done in Sec. 3.2.1, setup a simulation to verify that the block diagram given in Fig. 12 and its transfer function in Eq. 28 realize the same system. Do this by having the same input to each in SIMULINK and then compare the outputs. When comparing the outputs, you might want to find and plot the difference between the two outputs such as is done in Fig. 13. The difference should be small and only be non-zero due to small round-off errors in the computations.

Choose some non-zero value for A and k_r when performing the verification.

3.3.2 Step Response of Servos With and Without Rate Feedback

Let's compare the step response for the servos with and without rate feedback. Implement the block diagram given in Fig. 14 that inputs a step function to both types of servos and plots both step responses on the same plot. Using the values given in Sec. 3.1, start with simulating the servos using the follow extra settings.

$$A = 300$$
$$k_r = 0$$

With $k_r = 0$, there effectively is no rate feedback in either servo so both servos should give the same response. Verify that this is the case.

Now that you have verified that both servos function in the same way when $k_r = 0$, now try some values for k_r from about $k_r = 0.2$ to $k_r = 10$. Observe what happens as k_r is increased.

> **Report:** Your report should include a discussion of how increasing k_r affects the step response.

3.3.3 Selecting a damping ratio for the servo with rate feedback

In the previous section you observed how changing the value of k_r changes the behaviour of the step response by changing ζ. By choosing an appropriate k_r, you should be able to set the ζ of the system.

Take Eq. 31 that you determined and solve for k_r.

$$k_r = \tag{36}$$

Now that you have k_r as a function of ζ, choose a k_r such that the step response of the servo with rate feedback from the Sec. 3.3.2 is *critically damped*. Plot the step response of both the servo without rate feedback and the servo with rate feedback for this case.

You can check that your choice of k_r is correct by zooming in on your plot on the y-axis as much as you can near the final value. There should be no overshoot or oscillations in the response.

Also, to further check your choice of k_r, select a k_r that is slight smaller than the one you calculated (make k_r about 1%–5% smaller). Then find the step response in this case and zoom in on your plot on the y-axis as much as you can near the final value. You should see a small overshoot.

Report: Your report should include the calculated rate feedback constant k_r as well as a plot of the step response with and without (on the same plot) the rate feedback loop. Also, discuss what is observed when k_r is decreased by a small amount.

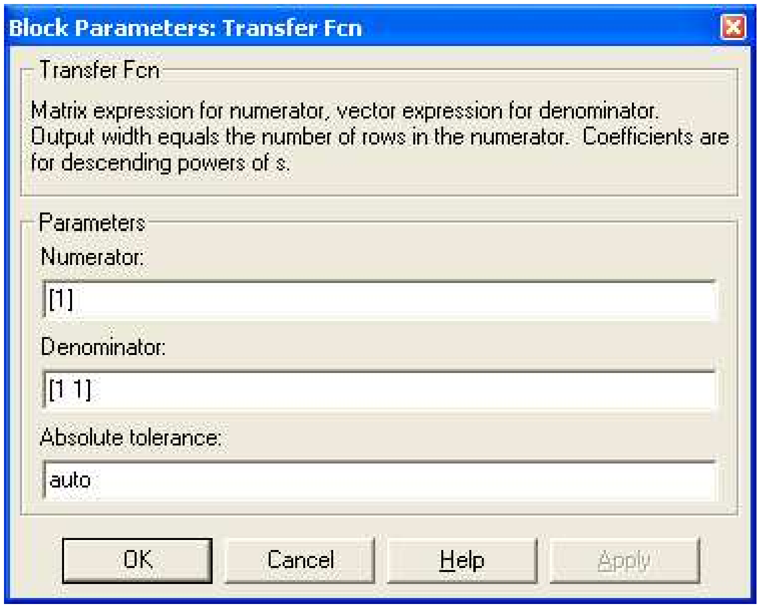

Figure 7: Block parameters for the `Transfer Fcn` block.

A SIMULINK/MATLAB Extras

A.1 Transfer Functions

Under the `Continuous` section of the SIMULINK Library Browser is the `Transfer Fcn` block. This block allows transfer functions to be modelled within SIMULINK.

To represent transfer functions, MATLAB and SIMULINK assume the following form for a differential equation.

$$a_N \frac{d^N y(t)}{dt^N} + \ldots + a_1 \frac{dy(t)}{dt} + a_0 y(t) = b_M \frac{d^M x(t)}{dt^M} + \cdots + b_1 \frac{dx(t)}{dt} + b_0 x(t) \tag{37}$$

Using this form for a differential equation, the corresponding transfer function $H(s)$ is in the form

$$H(s) = \frac{b_M s^M + \cdots + b_1 s + b_0}{a_N s^N + \ldots + a_1 s + a_0}. \tag{38}$$

You should notice that to represent this transfer function, you simply need to know the coefficients b_k and a_k in the numerator and the denominator, respectively. In MATLAB and in SIMULINK, these coefficients are represented by two vectors, one vector for the b_k coefficients in the numerator and another vector for the a_k coefficients in the denominator.

The `Transfer Fcn` block in SIMULINK accepts two parameters under its block parameters window as shown in Fig. 7. The parameters are a vector of the coefficients b_k for the numerator and a vector of coefficients a_k for the denominator. To give an example, consider the following transfer function.

$$H(s) = \frac{3s^2 + s + 7}{5s^3 + 8s + 16} \tag{39}$$

This transfer function would be represented with a numerator vector of `[3,1,7]` and a denominator vector of `[5,0,8,16]`. Notice in this example that there is no s^2 factor but the coefficient 0 must be included in the vector.

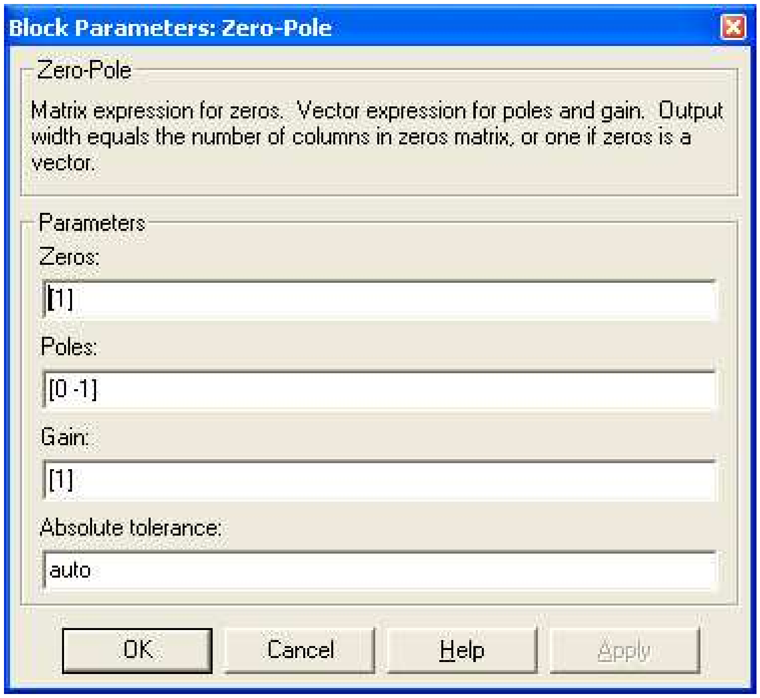

Figure 8: Block parameters for the `Zero-Pole` block.

To implement the transfer function block in Fig. 2, which is

$$H(s) = \frac{\Omega(s)}{T_m(s)} = \frac{1}{Js + b} \tag{40}$$

corresponding to Eq. 6, you would use the numerator vector `[1]` and the denominator vector `[J, b]`. Note that you can use either constants or MATLAB variable names in the vectors.

A.2 Pole/Zero/Gain Defined Transfer Functions

Another useful block in SIMULINK is the `Zero-Pole` block under the `Continuous` section of the SIMULINK Library Browser. This block also defines a transfer function, but unlike the `Transfer Fcn` block it defines the transfer function through its poles, zeros, and static gain.

Double-clicking on the `Zero-Pole` block gives its block parameters window as shown in Fig. 8. The poles and zeros are entered as vectors, while the gain is entered as a single value. For the transfer function defined by Eq. 6 we would have

$$H(s) = \frac{\Omega(s)}{T_m(s)} = \frac{1}{Js + b} = \frac{1/J}{s - (-b/J)} \tag{41}$$

such that the gain is $1/J$, the pole vector is simply `[-b/J]` since there is only one pole, and the zero vector is `[]` since there are no zeros. Be careful in this case not to put `[0]` for the zeros vector since that would indicate that there is one zero located at $s = 0$ (*i.e.*, that there is an s in the numerator, which is clearly not the case).

A.3 MATLAB `.m` Scripts

It is often convenient to use MATLAB script files to execute a list of MATLAB commands. MATLAB script files are simple text files, with the filename extension `.m`, in which you can place any MATLAB commands just as you would type them in the MATLAB `Command Window`.

For instance, you might want to create a file called `vars.m` containing

```
% Define variables for Simulink model of the servomechanism.
km=1.5275;
J=100;        % Moment of inertia of axle
b=100;        % Damping friction on axle
Ra=1;
N=12;         % Gear ratio
k1=12;
k2=k1;        % Use same gain for both potentiometers
B=km/(Ra*J);
C=(Ra*b+km*km)/(Ra*J);
```

to define the variables listed in Eq. 32 as well as in Eq. 17 and Eq. 18. Note that the "`%`" symbol indicates the start of a comment and the "`;`" symbol after a command simply gets MATLAB to suppress displaying any results from the command. After creating the file `vars.m`, simply type

```
cd <directory containing vars.m>
vars
```

in MATLAB's `Command Window` to run the script. You can then use the `whos` command to list the variables or type one of the variable names to display its value (*i.e.*, type `km` to display its value).

B SIMULINK Implementations of the Block Diagrams

B.1 SIMULINK realizations for the servo with position feedback.

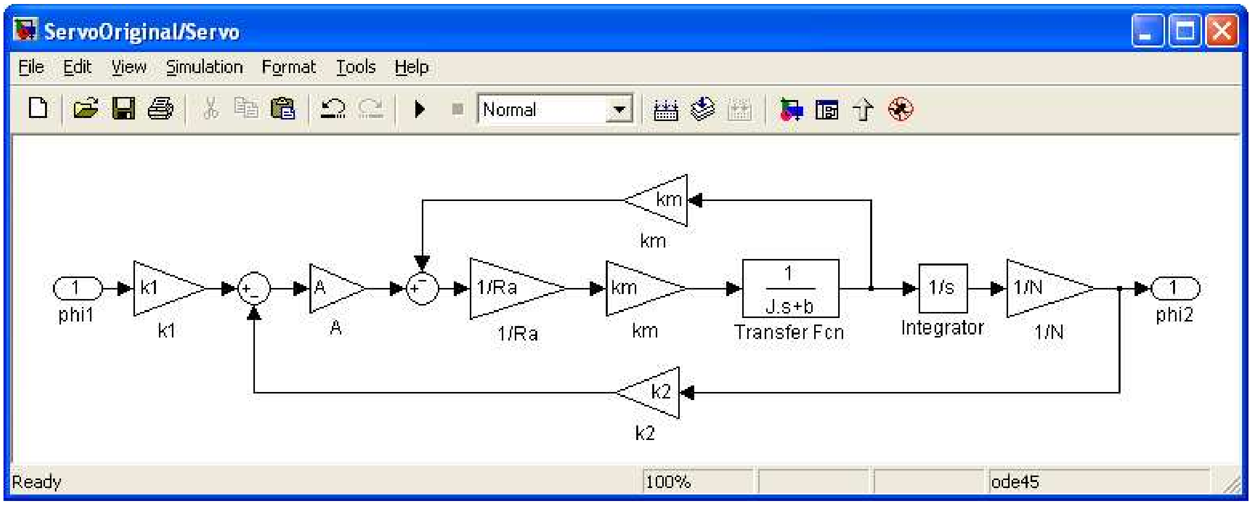

Figure 9: SIMULINK subsystem that realizes the servo with position feedback.

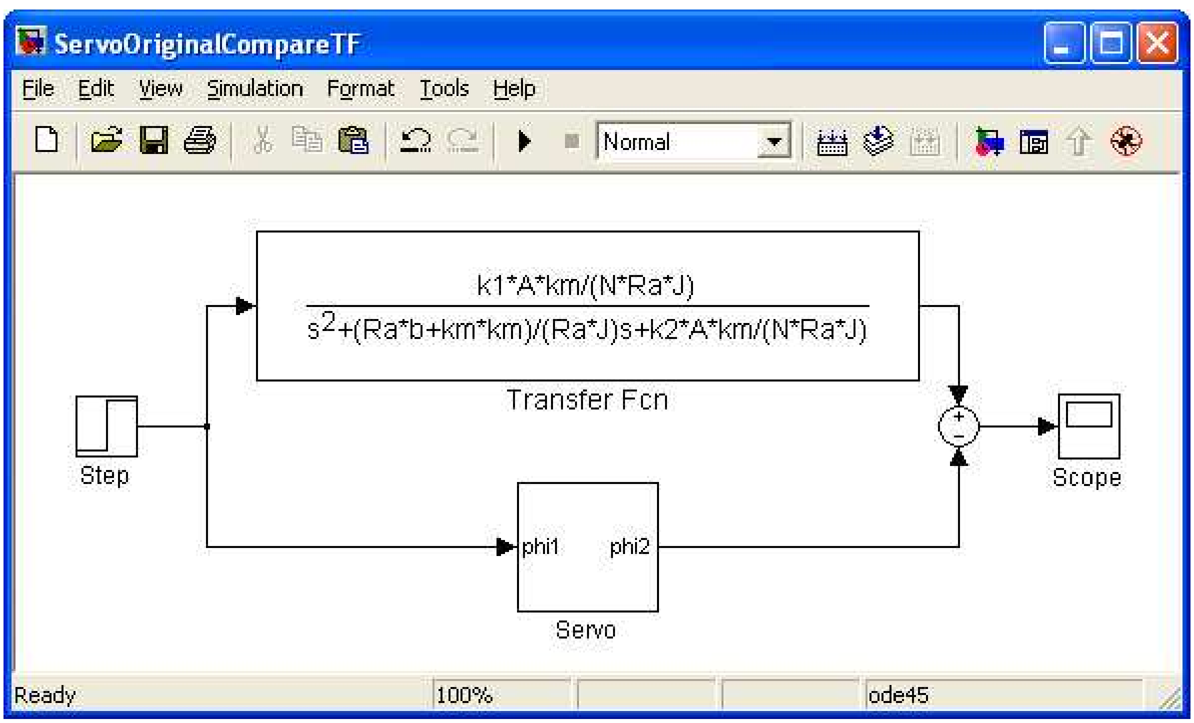

Figure 10: SIMULINK simulation that compares the step response of the block diagram for the servo with position feedback and its transfer function.

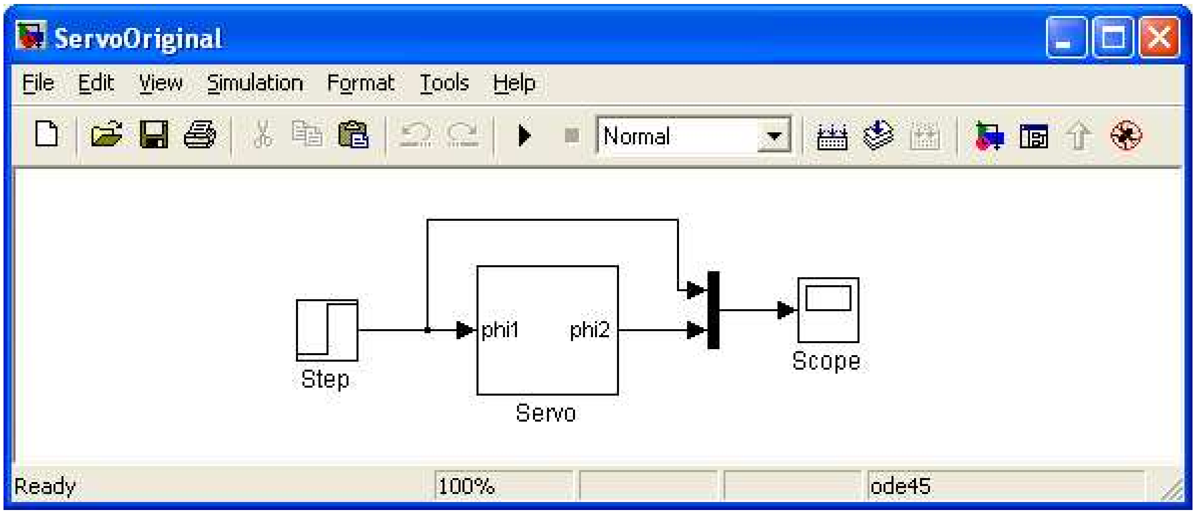

Figure 11: SIMULINK simulation of the step response for the servo with position feedback.

B.2 SIMULINK realizations for the servo with rate feedback.

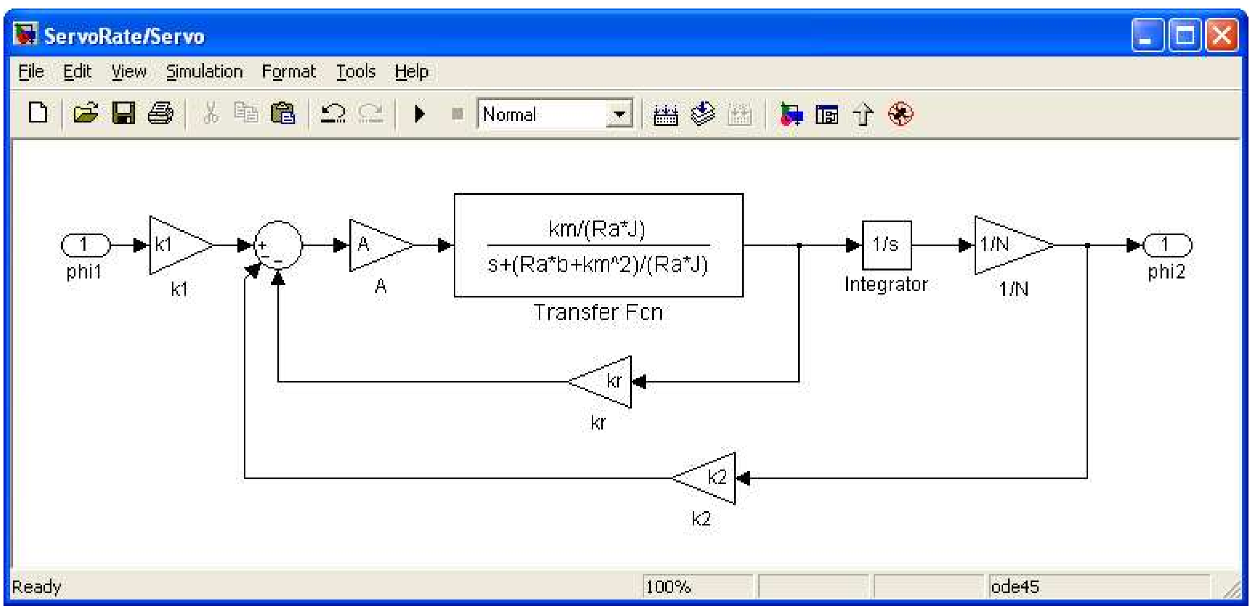

Figure 12: SIMULINK subsystem that realizes the servo with rate feedback.

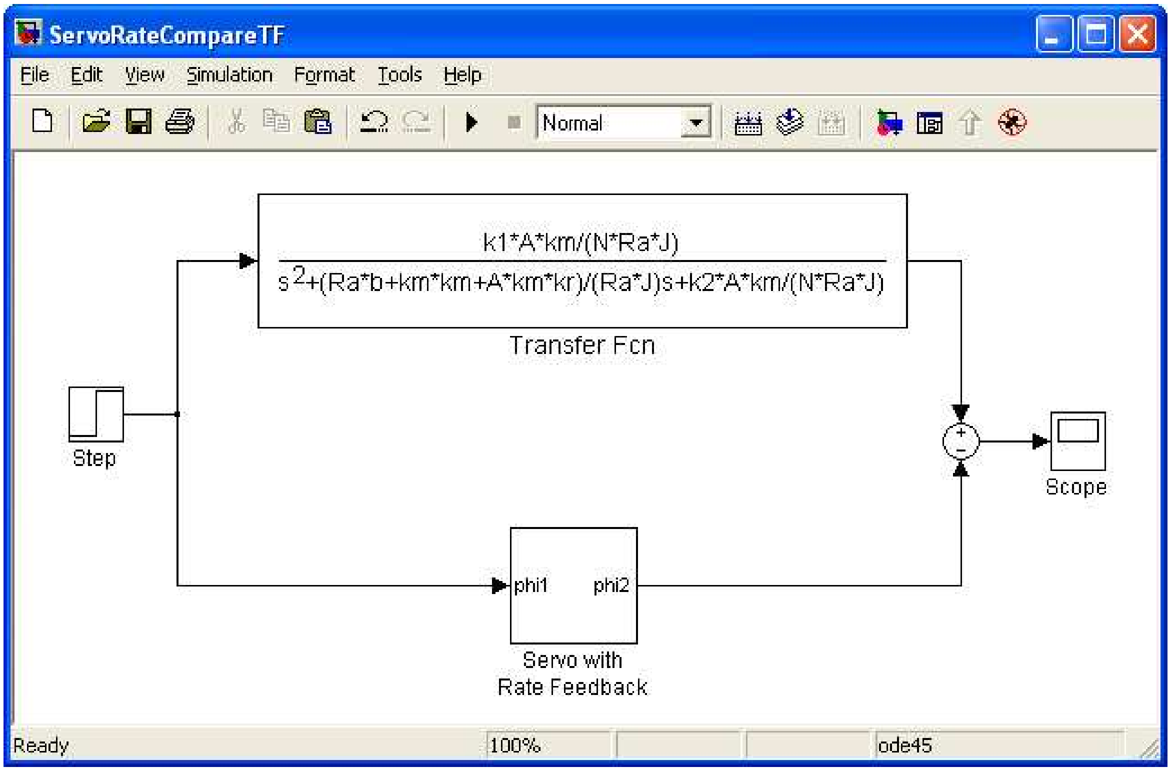

Figure 13: SIMULINK simulation that compares the step response of the block diagram for the servo with rate feedback and its transfer function.

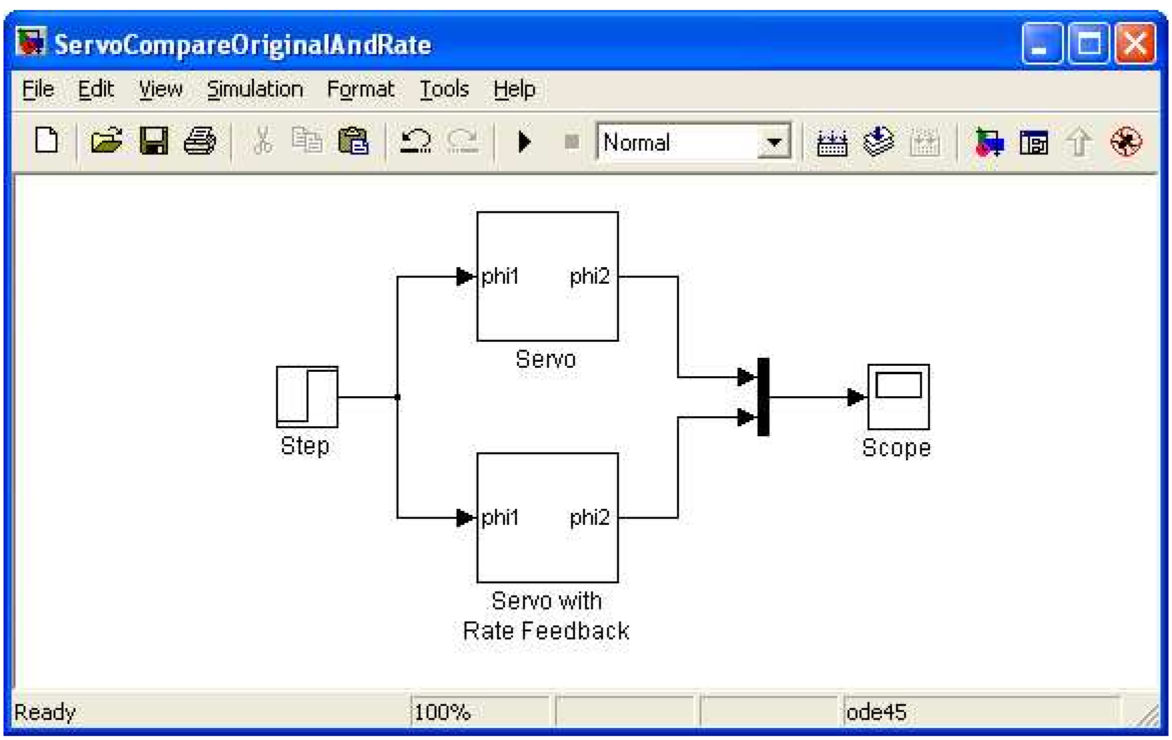

Figure 14: SIMULINK simulation of the step response through the servo with and without rate feedback.