MAE 3724 Systems Analysis
Fall 2019

Laboratory Experiment 5
Step Response of an Armature-Controlled DC Motor

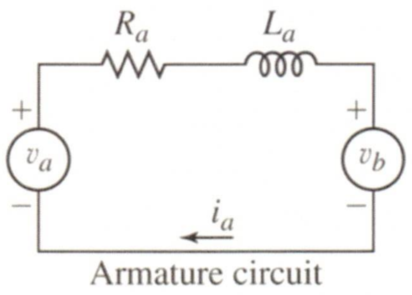
SAFETY: DO NOT STAND OR PUT YOU HAND IN THE PLANE of the ROTATING INERTIA (FOUR ARM FLYWHEEL ON THE DC MOTOR SHAFT).

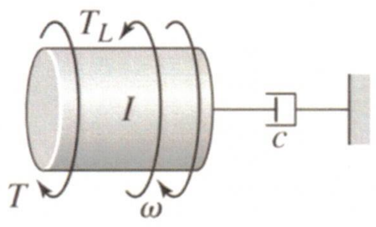
Background for Laboratory Experiment 5

The purpose of this experiment is to give you hands-on experience with an actual system of the type you have studied and modeled analytically, and how you can use the model and experimental data to determine unknown parameters.

In this experiment, you will measure the step response of an Armature-Controlled DC motor of the type you modeled in the Pre-Lab. The physical model is shown below. A specification sheet for the Armature-Controlled DC Motor that you will use in the laboratory is attached to this lab sheet. The motor has a built-in rotary encoder.

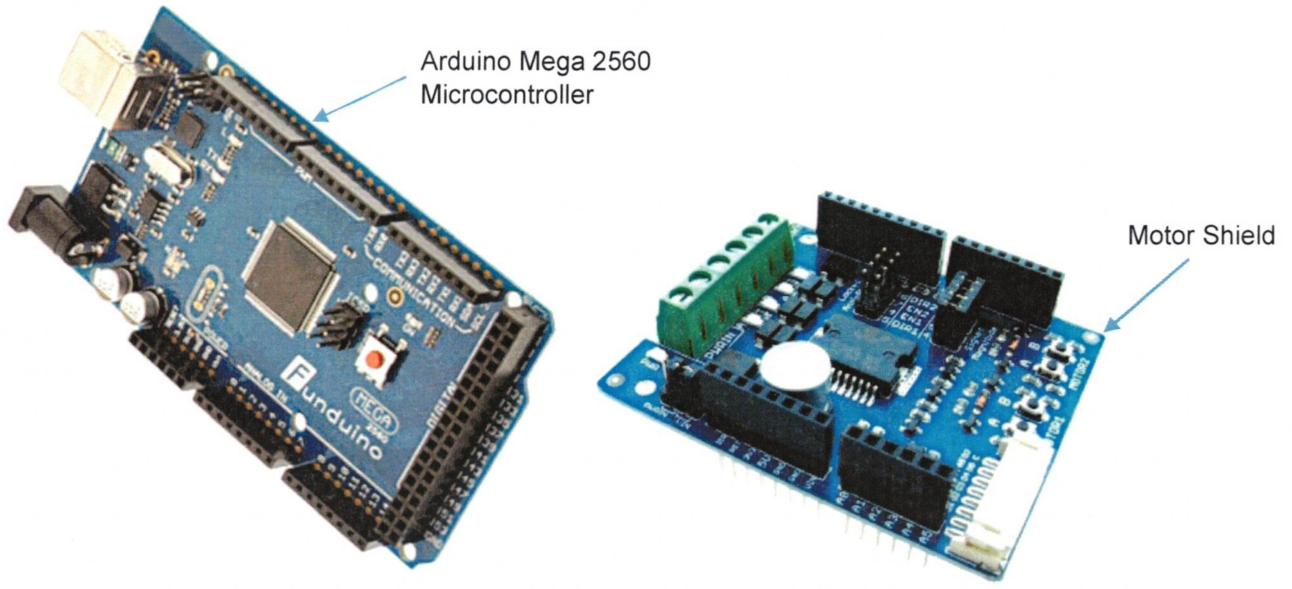
The values of the parameters for I and c that you used in the Pre-Lab will be different for the actual motor in this experiment. You will determine good estimates for these parameters from your experimental data and the motor data sheet (see Final Report).



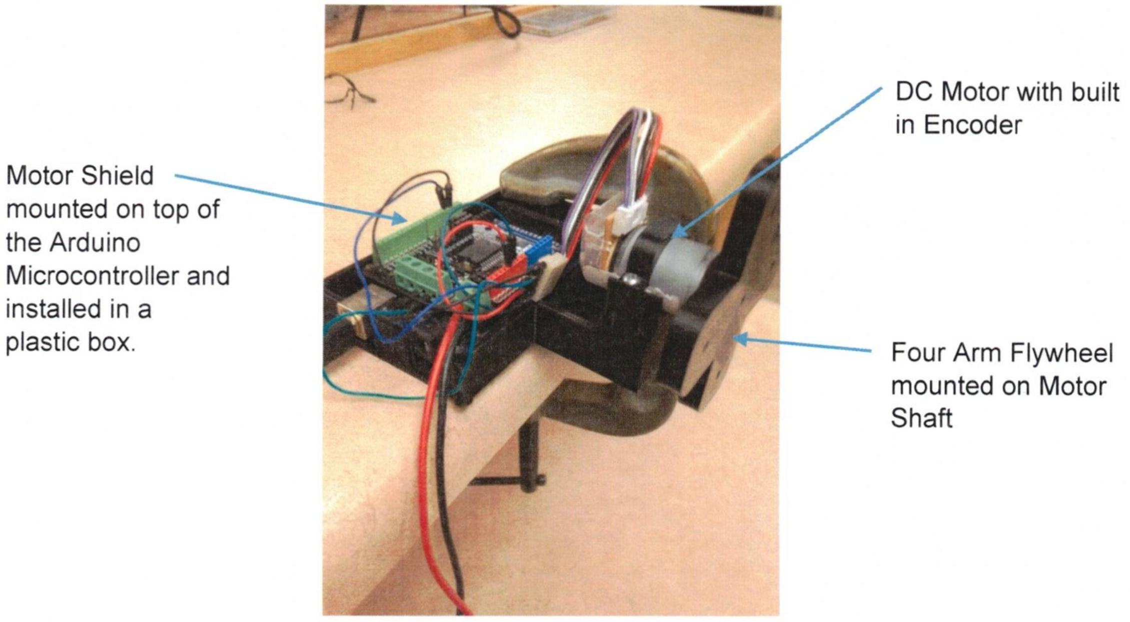


Setup

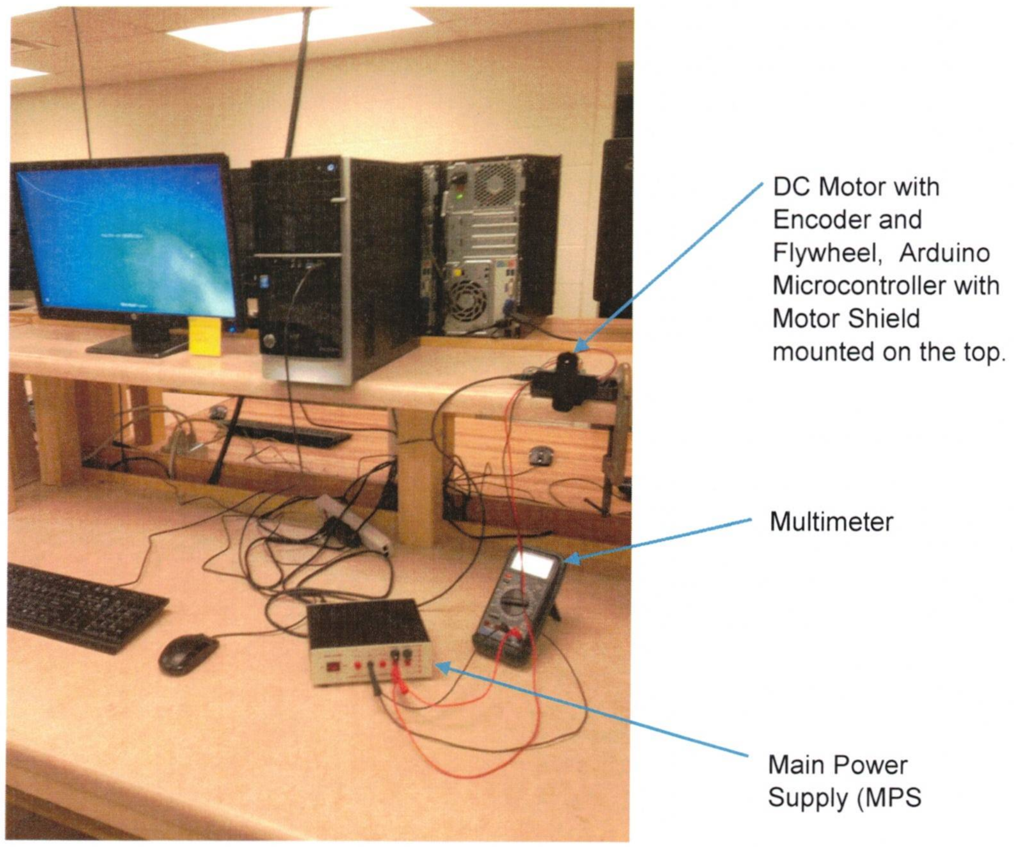
You will use an Arduino Mega 2560 microcontroller (see figure on next page) with a Motor Shield (see figure) mounted on the top to drive the Armature-Controlled DC Motor (see Figure below). The Arduino will process the data from Simulink and drive the Motor Shield. The Motor Shield drives the motor. The Arduino also acquires and processes the data from the encoder (built into the motor) and supplies it to Matlab and Simulink.



The DC motor is mounted in a plastic box made with a 3D printer. A four blade flywheel is mounted on the motor shaft to provide rotational inertia. Pennies can be placed (with sticky putty) into slots on the motor side of the flywheel to add inertia. Each group will be assigned a certain number of pennies, and therefore will have a certain rotational inertia (I). You may assume that $T_L = 0$ (Unless you grab the Flywheel !! **BUT, DO NOT DO IT.**)



The full set up is shown in the figure below. You will use the main computer for data acquisition and analysis. The program for the motor experiment is already installed and includes the use of Matlab for data acquisition and Simulink for control of the Arduino microcontroller. The output of the Motor Shield mounted on the Arduino drives the Armature-Controlled DC Motor. The output pulses from the encoder built into the motor are acquired and processed within the data acquisition system. A power supply provides the input voltage to the Arduino and to the motor. A multimeter is available to check voltages and resistances.


DC Motor with Encoder and Flywheel, Arduino Microcontroller with Motor Shield mounted on the top.
Multimeter
Main Power Supply (MPS

Procedure

The Arduino and Motor Shield are already set up for you.

Attach the USB cable already attached to the Arduino to the USB port on the computer.

Double check that the pennies provided at your station by the Lab Assistant are correctly installed to the slots in the four arm flywheel. Add the pennies in a symmetrical fashion to minimize unbalance.

Set the dial on the Multimeter to 20 V (DC).

Unplug the DC motor from the Multimeter. Keep JUST the multimeter attached to the Main Power Supply (MPS) using the black and red wires. (Black = 'GND', Red = '+ 15V DC')

Turn on the MPS and check the voltage with the Multimeter. It should be set to about 9 V. (can be slightly higher or lower than 9 V.) Do NOT exceed 10 V. This is NOT the voltage supplied to the motor. It is a maximum not to be exceeded. 8.65V

Turn off the MPS.

On the computer "Desktop" find the folder "Systems_Lab_5 ." Double click on the folder. (If unable to see, double check that you're logged in on the correct account).

Find the Simulink file "Open_Loop_Step_Response.slx". Double click on this file. The MATLAB window should pop-up.

You will see in the lower left corner of the MATLAB window the message "initializing."

The Simulink diagram that includes the block needed to study the system automatically comes up on the screen. Everything needed has been programmed for you so you can concentrate on the motor. However, you should observe all the elements in the Simulink diagram and computer display.

You will use three elements for this experiment: (1) "Constant" block (middle left on the Simulink diagram). This is the <u>actual</u> voltage supplied to the motor. (2) The Arduino programming button (on the bar in upper center of the display) - looks like a small Braille touch pad with three down arrows. (3) The "Plot Data Single" in the lower right on the Simulink diagram.

Double click on the "Constant" block and set the voltage value ('Constant value:') at 5 V. This will be the voltage applied to the motor circuit (v_a in the physical model represented by the circuit diagram on page 1). Double check that the "Sample time" is 0.01s. Click OK.

Click on the Arduino programming button (small Braille pad in the upper bar) and watch what is happening in the lower left corner of the computer display. The program is now compiling and building for the Arduino. You will see, "Building". When you see the message "Model successfully deployed to the Arduino Mega 2560", you are ready to run the experiment.

If the download is unsuccessful (error message pops up), close the message and try again. If the error continues, ask the Teaching Assistant for help.

Attach the Arduino to the MPS. (Black = 'GND', Red = '+ 15V DC')

Turn on the Main Power Supply (MPS). The motor should now run at a speed dependent on the input voltage (v_a) set in the "Constant" on the Simulink diagram.

<u>SAFETY</u>: <u>DO NOT STAND OR PUT YOUR HAND IN THE PLANE of the ROTATING INERTIA (FOUR ARM FLYWHEEL ON THE DC MOTOR SHAFT).</u>

Click on “Plot Data Single” on the Simulink diagram. A COM port window will pop up.

Double check that the COM port is correct for your computer. Set the number of samples to plot to 1000. Click OK on the COM Port pop up. (If a figure does not automatically pop-up, check the MATLAB Command Window. If the default port shows “is not available”, reclick “Plot Data Single” and change the “Enter COM port to collect data” to a port that is available.)

You will see Figure 1 pop up (a Matlab screen) and observe plotting in red on the screen. Click “Auto Scale” and observe that the response does not start at zero.

Turn off the MPS and observe what is happening on Figure 1. (The scales on Figure 1 do not have meaning – they will be corrected later). When the motor speed goes down to zero (but you cannot see the zero), click on the “Auto Scale” and observe that the plot now starts at zero.

Turn on the MPS and watch the plot on the screen. The plot now should start at zero and slowly move to the left. (If the plot does not start at zero, turn off the MPS, wait for the motor speed to reach zero and try again.)

When the plot is almost fully finished (i.e., steady-state speed is established for a few seconds) and while you can still see the full plot, click “Stop” on Figure 1. This “freezes” the data collection of rotational speed vs time. Do not worry about the scales on Figure 1. They will be corrected later. Your data has now been acquired and put into the workspace of Matlab.

Find the Matlab icon on the task bar, click on it and see several possible screens. Click on the far left screen (Matlab main interface). In the left panel under “Name”, find the file “Plotter.m.” Double click on this file and an “Editor” window will open inside the Matlab interface. Find the green “Run” button (triangle) at the top of the Matlab interface screen.

Click on “Run.” A new figure appears with the step response data. The scales are now corrected to motor speed in rad/sec and time in seconds. This action also created an Excel file with all the data.

(If an error message appears, most likely your data did not start at zero after you turned the MPS back on. Record the data again. Ask the Lab Assistant for help if needed.)

Find “Excel Data” in the left panel under “Name”. Right click and a window will pop up. Click on “Open Outside MATLAB.” The Excel file should now be open. There are two columns – first column is time (s) and the second column is rotational speed (rad/s). Label the columns appropriately.

Save and email your data to yourself. You will need to plot it yourself for the Post-Lab.

After emailing your data, delete it from the Laboratory Computer desktop.
Do not log off of the current account (see note on the bench).

MAE 3724 Systems Analysis

Fall 2019

Laboratory Experiment 5
Step Response of an Armature-Controlled DC Motor

Final Report for Lab 5 (Each individual must submit a Final Report)

SHOW ALL YOUR WORK ON THIS AND THE FOLLOWING PAGE. Scan these pages and the requested plots and upload them into the Brightspace Dropbox for Lab 5.

The most important part of the Final Report is to compare the measured data with the calculated data based on the model. But you must first put the correct parameters into your model.

Recall the transfer function developed in the Pre-Lab:

$$T(s) = \frac{\Omega(s)}{V_a(s)} = \frac{\dfrac{K_T}{cR_a + K_bK_T}}{\left[\dfrac{R_aI}{cR_a + K_bK_T}\right]s + 1}$$

Use the experimental data to estimate the following ***(SHOW ALL YOUR WORK)***:

a) *[2 pts]* Steady-state value of the motor rotational speed, ω_{ss}.

$\omega_{ss} =$ 163.37 rad/s

b) *[2 pts]* System time constant τ (use the 63% Rule).

$\tau =$ 0.79 s

c) *[2 pts]* Damping coefficient, c. For these calculations $V_a = 5\ V$, and the parameters $K_T, K_b, and\ R_a$ are the same as in the Pre-Lab. Hint: set $s = 0$ in the transfer function above and interpret in the time domain.

$$\omega_{ss} = \frac{K_TV_a}{cR_a + K_bK_T} \rightarrow c = \frac{1}{R_a}\left[\frac{K_TV_a}{\omega_{ss}} - K_bK_T\right]$$

$c =$ 7.724 E-6 $\frac{Nms}{rad}$

d) *[2 pts]* Rotational inertia I. Hint: We know from the transfer function derived in the Pre-Lab that

$$\tau = \frac{R_a I}{cR_a + K_b K_T} \rightarrow I = \frac{1}{R_a}\left[\tau(cR_a + K_b K_T)\right]$$

$I =$ 47.82 E-6 Kg m^2

e) *[3 pts]* Update your model with the new parameters identified from the experiment (new I and c). **Prepare an overlay of the measured and predicted step responses and attach it to your report.** *(Use a legend to distinguish the responses.)*

f) *[2 pts]* How do the predictions compare with the measurements?

The model closely approximates reality

g) *[3 pts]* Plot just the first $\mathbf{0.3}$ ***seconds*** of the predicted and measured responses to better see the initial slope of the data. **Attach this zoomed-in plot to your report.**

h) *[2 pts]* For the plot in part g), look at the slope around $t = 0\,s$. Is there a higher order effect? If so, explain the origin?

Yes, there is a "lag" in the actual system when compared to the simulation

i) *[2 pts]* Was the assumption that $L_a \approx 0$ a good assumption? Why or why not?

Yes, it reduces the complexity of the system while remaining accurate enough through the rest of the model simulation.

Final Evaluation (for feedback purposes, will not affect grade)

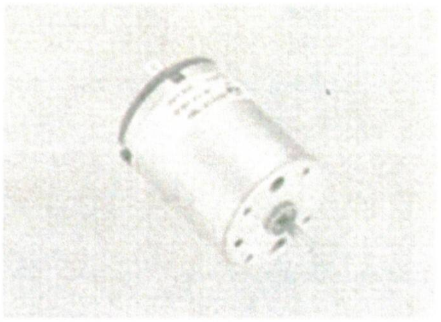
1) What did you learn from this experiment?

2) What could we do to make the laboratory experiment a better learning experience?

DC Mini-Motors
M25N-1 Series

APPLICATIONS

Printer

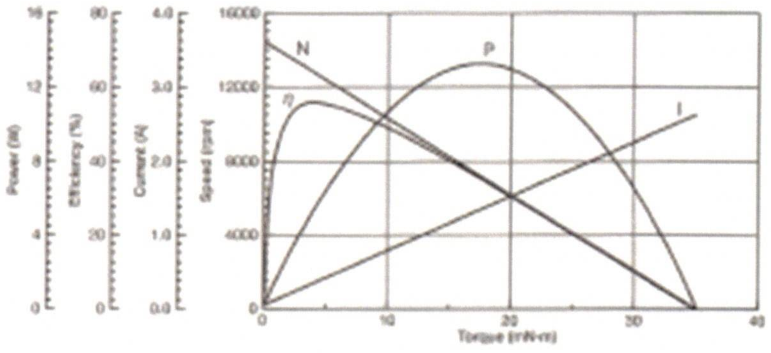


SPECIFICATIONS

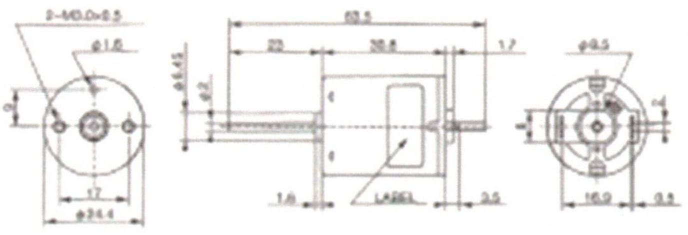
Items	Specifications
Rated Voltage	28.0V
Voltage Range	22.0-34.0V
Rated Load	-
No Load Speed	14,400rpm
No Load Current	100mA or less
Starting Torque	35mN·m
Rotation	CW/CCW

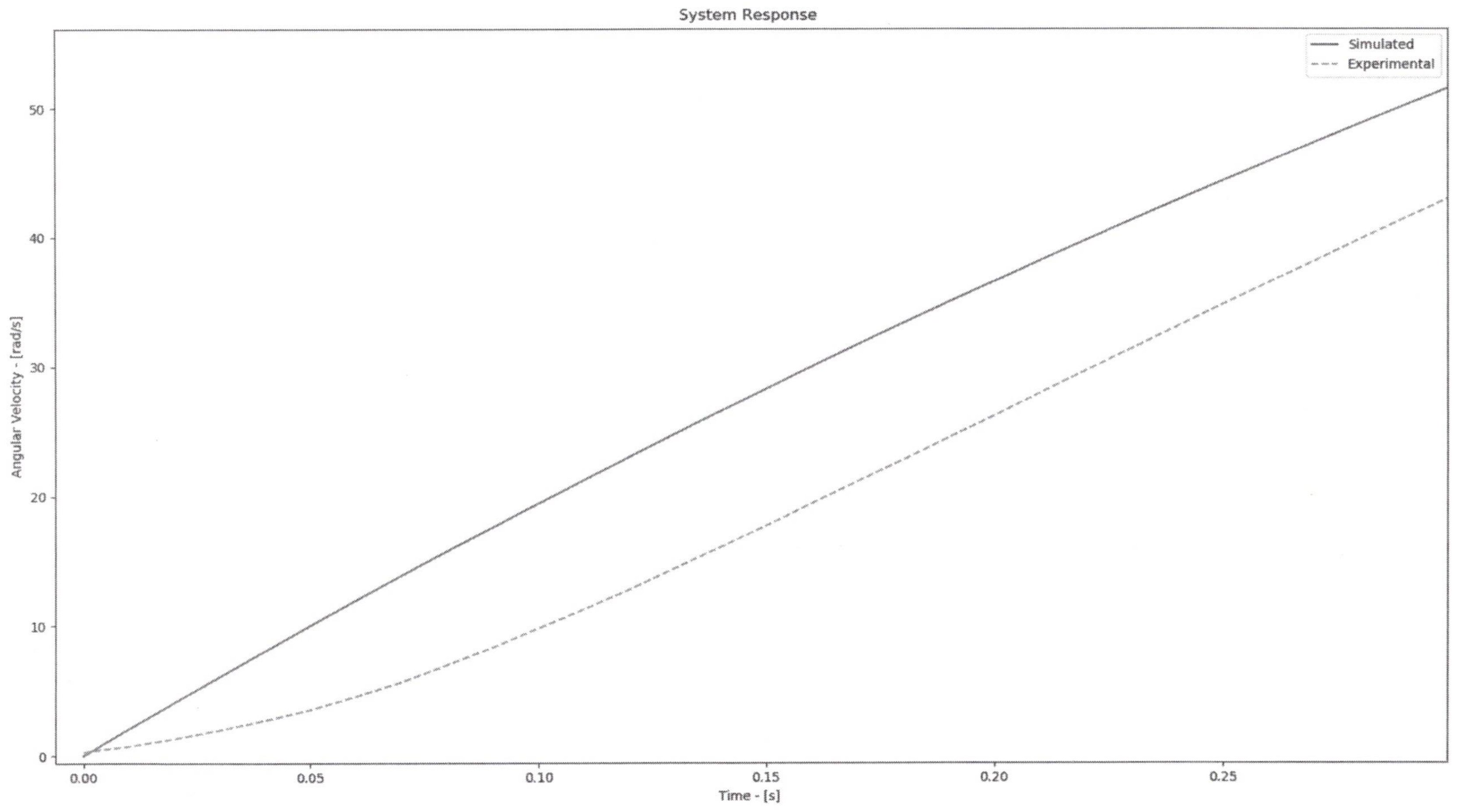
*Characteristics and the shaft length can be customized. Encoder is available as options.

CHARACTERISTICS



DIMENSIONS




System Response
Simulated
Experimental
Angular Velocity - [rad/s]
Time - [s]
0
10
20
30
40
50
0.00
0.05
0.10
0.15
0.20
0.25