Lab 1: Digital PID Speed Control of a Turntable

Group 1

Jeremy Nielsen

Tara Boyle

Scott Miller

September 16, 2019

MEC 411 Control System Analysis and Design

Department of Mechanical Engineering

Stony Brook University, College of Engineering and Applied Sciences

Contents

I Abstract

The objective of this experiment was to implement a proportional-integral-derivative (PID) feedback loop control system to control the rotational speed of a turntable. Several system parameters were adjusted to meet the desired performance specifications. Theoretical results were calculated using MATLAB and compared the experimental data. Experimental results were gathered using software with a data acquisition (DAQ) system to measure parameters of an electrical circuit which was constructed to model a given block diagram. The findings showed that the values of $K_c = 0$, $T_i = 5.0 \times 10^{-4}$, and $T_d = 5 \times 10^{-5}$ led to a percent overshoot of 2.13%, a settling time of $T_s = 0.065$ seconds, and a rise time of $T_r = 0.0799$ seconds for the PID controller. These values fall within the desired performance specifications which means the PID speed control system adequately regulated the rotational speed of the turntable. This lab was also performed to demonstrate knowledge of closed loop control systems.

II Introduction

In order to properly analyze and design a control system, the first step is to understand the underlying theory governing the system [1]. Specifically, this experiment consists of adjusting the parameters of a PID feedback loop to control the rotational speed of a turntable to achieve a set of given performance specifications. A waveform generator was used to create a square wave with an amplitude of 1V (2V peak to peak) at a frequency of 1 Hz. The waveform generator is connected to a DAQ system so that the signal can be viewed in the LabView control software. An input signal of 12V to the turntable leads to an unloaded rotational speed of 1000 RPM [2]. However, the input signal of the waveform generator steps the voltage down to 1V. The rotational speed of the turntable varies proportionally to voltage, so the output becomes a square wave with a magnitude of 83.3RPM, which the PID controller must try to match. This happens by recording the tachometer sensor signal which is related to the motor speed.

A tachometer functions by detecting the frequency of voltage pulses produced by the spinning of a magnet that is attached to the shaft of the motor [3]. The tachometer voltage gradient is given to be 0.52 V/KRPM. This means that the output signal is scaled so that 0.52V is the output for 1000 RPM. The signal goes through a non-inverting proportional operational amplifier (op-amp) to provide a gain of 12V/0.52V so that 1000 RPM now creates a 12V signal. This gain is accomplished in the circuit by adjusting a potentiometer to the proper resistance to match the gain ratio. The derivation for this resistance value relies on the fundamental operating principles of operational amplifiers as follows:

$$V_{out} - I_2 R_2 - 0 - V_{in} = 0 \tag{1}$$

So the R_2 resistance that makes V_{out} equal to 12V, or the K_t value, must be determined. Therefore, the expression for the current across the potentiometer is necessary:

$$I_2 = \frac{V_{out} - V}{R_2} \tag{2}$$

The current across the resistor is also necessary:

$$I_1 = \frac{V_{in} - 0}{R_1} \tag{3}$$

Finally, the currents are set to be equal to each other since this op-amp is assumed to behave ideally:

$$V_{out} = V_{in}(1 + \frac{R_2}{R_1}) \tag{4}$$

Then the potentiometer resistance, R_2, was solved to be $23.1k\Omega$. This $23.1k\Omega$ resistor is connected from the negative terminal of the op-amp to its output, and a $1k\Omega$ resistor to produce a gain ratio of 23/1, which is close to the desired gain of 12/0.52. The amplifier was non-inverting because the voltage input was connected to the non-inverting input terminal, so the output voltage maintains the same sign as the input. This op-amp in the circuit was used to convert the output of the system into the same sign as the system's input.

This signal goes to the DAQ and is the PID Process Variable. This is important because measuring the output signal is the first step in being able to adjust it to be closer to a desired output. The PID controller measures the difference between this output and the target signal generated by the waveform generator to find the error in the output. Three parameters - the proportional gain (K_c), integration time (T_i), and derivative time (T_d) are used to control how the controller adjusts to reach the correct value. This process happens each time the input signal switches polarity. The LabView program has knobs that can be turned to adjust the values of the parameters. The feedback signal is then sent to another operational amplifier with a push-pull follower modifying the output which controls the motor speed with the improved output. A power supply provides ±12V for the push-pull follower and a common ground for the waveform generator, DAQ, and DC motor. In terms of the actual control loop, there must be an equation which relates the inputs of the system to its outputs. This transfer function can be derived as follows:

$$P_1 = G_c(s) * G(s) \tag{5}$$

$$L_1 = -k_t * G_c(s) * G(s) \tag{6}$$

$$\delta = 1 - L_1 = 1 + k_t * G_c(s) * G(s) \tag{7}$$

$$\delta_k = 1 \tag{8}$$

$$T(s) = \frac{V(s)}{\omega(s)} = \frac{\sum P_k * \delta_k}{\delta} = \frac{G_c(s)G(s)}{1 + K_t G_c(s)G(s)} \tag{9}$$

The closed loop transfer function is:

$$T(s) = \frac{\omega(s)}{V(s)} = \frac{G_c(s)G(s)}{1 + K_t G_c(s)G(s)} \tag{10}$$

where

$$G(s) = \frac{K_m}{(L_a s + R_a)(Js + b) + K_b K_m} \approx \frac{K_m}{R_a (J_S + b) + K_b K_m} \tag{11}$$

and

$$G_c(s) = K_c \left(1 + \frac{1}{T_i s} + T_d s\right) \tag{12}$$

$G(s)$ is a negative feedback loop composed of the:

$$\text{Armature} = \frac{K_m}{L_a s + R_a}, \tag{13}$$

$$\text{Load} = \frac{1}{J_S + b}, \tag{14}$$

and

$$\text{Back EMF} = K_b. \tag{15}$$

The paramters of the DC motor, tachometer, and load are given to be:

$$K_m = 16.2\,\mathrm{OZ - IN/A}, \tag{16}$$

$$R_a = 11.5\,\Omega, \tag{17}$$

$$L_a = 0, \tag{18}$$

$$J = 2.5\,\mathrm{OZ} - \mathrm{IN}^2, \tag{19}$$

$$b = 0, \tag{20}$$

$$K_b = 12\,\mathrm{V/KRPM}, \tag{21}$$

and

$$K_t = 12\,\mathrm{V/KRPM}. \tag{22}$$

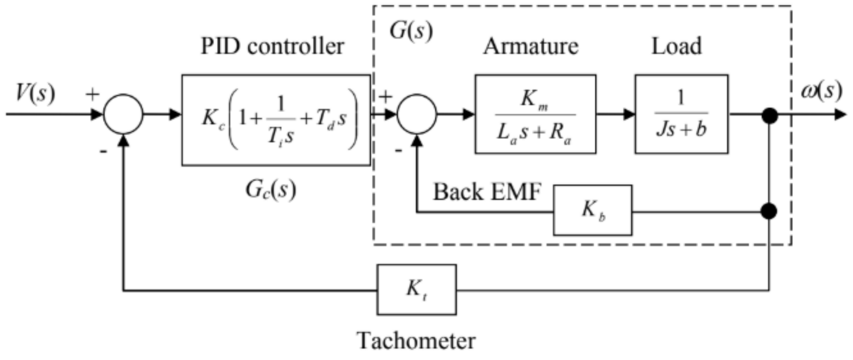


Figure 1 PID Speed Control System Block Diagram

Once a clear understanding of the control system is established, there must be a figure of merit to precisely define whether or not the system is successful when subjected to a unit step input. In the case of this experiment, the design specifications were given to be a percentage overshoot (PO) less than 10%, a settling time (T_s) to within 2% of the final value in less than 500ms, and rise time (T_r) less than 200ms. PO, is calculated as follows:

$$P.O. = 100e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}}. \tag{23}$$

Where ω_{max} is the peak ω value and $\omega_{settled}$ is the settled output [4]. Next, the settling time is the time taken for the function value to settle within 2% of the expected value and it is represented as follows:

$$T_s = \frac{ln(2/100)}{\omega_n \xi}. \tag{24}$$

Rise time, T_r, is defined as:

$$T_r = \frac{\pi - \cos^{-1}(\xi)}{\omega_n \sqrt{1-\xi^2}}. \tag{25}$$

These mathematical relationships give a clear, complete, and thorough understanding of the principles of the experiment.

III Experimental Procedures

The experiment was performed with the following equipment: an Agilent E3630A power supply, an Agilent 33220A waveform generator, a National Instruments DAQ system, a Fairchild TIP 122 NPN darlington transistor, a Fairchild TIP 127 PNP darlington transistor, multiple operational amplifiers, a DC motor with a tachometer, a turntable, a 100K potentiometer, a breadboard, wires, resistors, capacitors, and diodes. Using the equipment, the turntable speed control PID system was constructed based on the circuit diagram provided (Figure 2). The potentiometer was set to 23.1 kOhms to normalize the output to the input. The function generator was used to create a square wave with an amplitude of 1 V (2 V peak to peak) at a frequency of 1 Hz. A LabView program was used as control software to adjust the integration time (T_i), derivative time (T_d), and proportional term (K_c) of the system. This control software was also used to record the experimental waveforms resulting from the constructed control system. A closed loop feedback system with a proportional (P) controller was made by setting T_d and T_i to zero. K_c was adjusted using the control software so that the turntable speed was as close to the input as possible while maintaining system stability. All system parameters for the P controller were recorded along with the input and output waveforms using the LabView program. The proportional-integral (PI) controller terms, proportional term (K_c) and integral time (T_i), were determined in a similar manner as described for the P controller. However, both K_c and T_i were tuned simultaneously to make the turntable speed match the input as closely as possible while maintaining system stability. All systems for the PI controller were recorded along with the input and output waveforms using the LabView program. The proportional-integral-derivative (PID) controller proportional term (K_c), integral time (T_i), and derivative time (T_d), were adjusted in LabView until the turntable speed was as close as possible to the input while maintain system stability. Additionally, the PID controller output needed to meet the design specifications of having a percent overshoot less than 10%, settling time to within 2% of the final value in less than 500ms, and a rise time of less than 200ms. Once all of the criteria were satisfied, the PID variables and waveforms were recorded. The recorded data for each control system were exported to Microsoft Excel. The three sets of experimental results for a P, PI, and PID controller were compared to the theoretical results computed by simulating the respective systems using a MATLAB program.

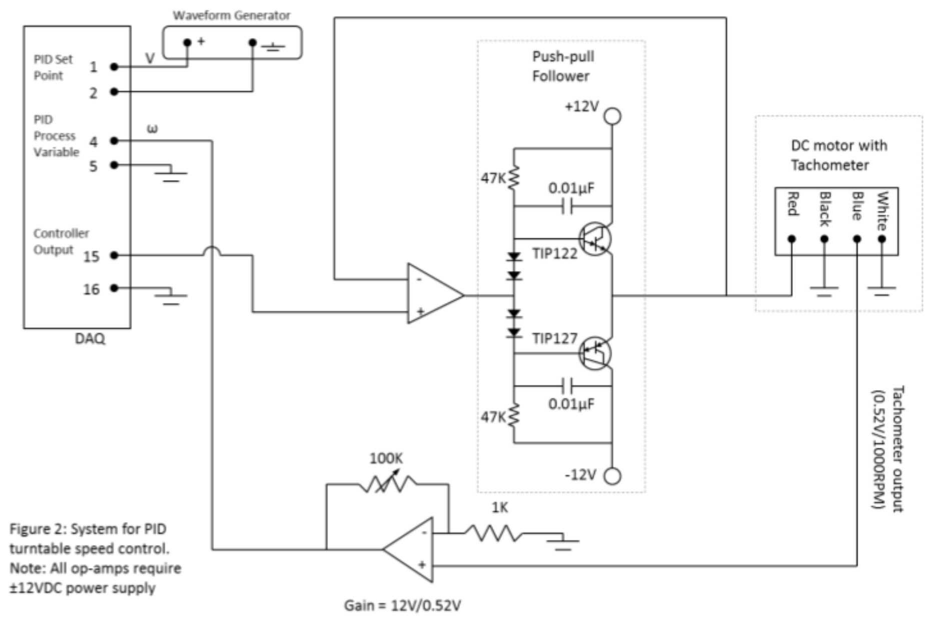


Figure 2 PID Speed Control System Circuit Diagram

IV Results

While attempting to calibrate the proportional controller by increasing K_c to emulate the square wave input signal, instability was observed at a value of 2.2. This instability occurred long before K_c could be increased enough for the output to reach anywhere near the accepted value. The K_c value was decreased until stability was observed again, this occurred when K_c was equal 2.0 as reflected in the Controller Calibration Table (Table 1). Due to the fact that the signal experienced instability before approaching the desired square wave, there is a large gap between the desired and actual outputs as observed in the experimental P controller input and output speed versus time graph (Figure 3). When a T_i value of 4.5×10^{-4} minutes was reached, the output signal of the PI controller was observed to be within an acceptable range to the desired square wave signal. This can be observed in the experimental PI controller input and output speed versus time graph (Figure 4). The ideal value is always within an acceptable range of error which is calculated in error analysis (Section V). The can be observed graphically to be within the specifications of having a rise time below 10%, a settling time to within 2% of the final value less than 500 ms, and a rise time is less than 200 ms (Figure 4). While the exact values for these specifications were not analyzed, since the final product was to be the PID controller, success for the PI controller would give an approximated value for K_c and T_i for the PID controller. While calibrating the PID controller as the final product of this procedure, the K_c and T_i parameters were kept to as close to the values used in the PI controller as possible, as the PI controller functioned within specifications. The values of for the variables were K_c was equal to zero, T_i was 5.0×10^{-4} minutes, and T_d was $5X10^{-5}$ minutes (Table 1). With these variables set, the functional PID control settings were reached. The PID controller output was an improvement on the PI controller in that it cut down on the rise time observed in its graph (Figure 5). The experimental rise time, settling time, and percent overshoot were all determined graphically from the experimental data (Figure 5). The rise time, T_r, was observed to be 0.0799 seconds from the start of the waveform, the error of which is calculated in the error analysis (Section V) along with the rest of the associated uncertainty values for each value presented in Results. The settling time, T_s, was observed to be 0.089 seconds from the start of the waveform. Lastly, the peak of the controller was determined to be 86.2 RPM, 86 milliseconds from the start of the waveform. With a final settling point of 84.4 RPM the percent overshoot, $P.O.$, is calculated as follows:

$$P.O. = 100(\frac{\omega_{max} - \omega_{settled}}{\omega_{settled}}). \tag{26}$$

Where ω_{max} is the peak ω value and $\omega_{settled}$ is the settled output. The percent overshoot is calculated below while the uncertainty is calculated in the error analysis (Section V):

$$P.O = 100(\frac{86.2 - 84.4}{84.4}) = 2.13\%. \tag{27}$$

MATLAB was used simulate the outputs of the P, PI, and PID controller. The code used to produce the simulated outputs can be found the in the appendix (Section IX)(Figures 1A-3A). The code utilized the transfer function Eq.(10) by substituting for $G(s)$, Eq.(11), and $G_c(s)$, Eq.(12). This results in the following output formula after solving for $\omega(s)$:

$$\omega(s) = \frac{K_m K_c T_d s^2 + K_m K_c s + \frac{K_c K_m}{T_i}}{s(K_m K_c T_d s^2 + (K_m K_c + 2R_a J)s + \frac{K_c K_m}{T_i} + 2K_b K_m)}, \tag{28}$$

where K_m is the torque constant, K_b is the back EMF constant, R_a is the DC armature resistance, K_t is the tachometer output ratio, J is the internal moment of the motor, T_i is the integration time, T_d is the derivative time, and K_c is the PID grain parameter. All of the aforementioned constant values are documented in the constants table (Table 2). This is converted as a function of time using the an inverse Laplace transform, which was accomplished by the ilaplace command in MATLAB. The results of the ilaplace command are as follows:

$$\omega(t) = 1/Kt - JR_aT_i e^{-t\frac{K_bK_mT_i+K_cK_mK_tT_i}{2JR_aT_i+2K_cK_mK_tT_dT_i}} *$$
$$(\cos(t * \frac{\sqrt{(K_b^2K_m^2T_i/4)-K_c^2K_m^2K_t^2T_d+(K_c^2K_m^2K_t^2T_i/4)+(K_bK_cK_m^2K_tT_i/2)-JK_cK_mK_tR_a}}{T_i^{1/2}(JR_a+K_cK_mK_tT_d)}) -$$

$$T_i^{1/2}\sin(t\frac{\sqrt{(K_b^2K_m^2T_i/4)-K_c^2K_m^2K_t^2T_d+(K_c^2K_m^2K_t^2T_i/4)+(K_bK_cK_m^2K_tT_i/2)-JK_cK_mK_tR_a}}{T_i^{1/2}(JR_a+K_cK_mK_tT_d)}) *$$

$$\frac{(\frac{K_bK_mT_i+K_cK_mK_tT_i}{2JR_aT_i+2K_cK_mK_tT_dT_i}-\frac{K_b*K_m}{J*R_a})(JR_a+K_cK_mK_tT_d)}{\sqrt{(K_b^2K_m^2T_i/4)-K_c^2K_m^2K_t^2T_d+(K_c^2K_m^2K_t^2T_i/4)+(K_bK_cK_m^2K_tT_i/2)-JK_cK_mK_tR_a}})/(K_t(JR_aT_i+K_cK_mK_tT_dT_i))$$

.(29)

It should be noted that both J and K_m are converted to metric for this calculation, T_i and T_d are converted to seconds and K_b and K_t are converted to V/RPS in order to ensure t is represented as seconds.

The MATLAB code simulated the behavior of the experimental P controller output within its uncertainty for the first half of the wavelength, which was calculated in the error analysis (Section V). However, the simulation failed to remain within accepted uncertainty ranges for the negative half of the P wavelength, as observed in the MATLAB Simulated and Experimental P Controller Input and Output Speed Versus Time (Figure 6). The MATLAB code simulated the behavior of the experimental PI controller output within its uncertainty with the exception of the overshooting section found around the times values of 0.9 and 5.8 seconds, as observed in the MATLAB simulated and experimental PI controller input and output speed versus time graph (Figure 7). Lastly, the MATLAB code simulated the output of the PID controller correctly with the only exception being the negative peak, as observed in the MATLAB Simulated and Experimental PID Controller Input and Output Speed Versus Time (Figure 8). From this simulation theoretical values for rise time $T_{r-theoretical}$, percent overshoot $P.O._{theoretical}$, and settling time $T_{s-theoretical}$ are graphically obtained. From the simulated data the following information was recorded: $T_{r-theoretical} = 0.06 \pm 0.0002$ seconds, a peak of 90.3985 RPM, and $T_{s-theoretical} = 0.1543 \pm 0.0619$ seconds. Using the $P.O.$ equation Eq.(26) and a settled signal of 83.3 RPM, the percent overshoot can calculated to be $8.52\% \pm 0.37$. The signal behavior of the PID controller show that the theoretical signal and the experimental signal are within the specification of our PID design parameters [3].

Table 1 Experimental Proportional, Integral and Derivative Time Terms for all Experimental Controllers

	K$_c$	T$_i$	T$_d$
P	2	0	0
PI	2	$4.5X10^{-4}$	0
PID	2	$5.0X10^{-4}$	$5X10^{-5}$

Table 2 Parameters of the DC motor, the Tachometer, and the Load [2]

Parameter	Value
K$_m$	16.2 OZ-IN / A
R$_a$	11.5 Ω
L$_a$	0
J	2.5 OZ-IN^2
b	0
K$_b$	12 V / KRPM
K$_t$	12 V / KRPM

Table 3 Signal Behavior of PID controller

	Specification Limit	Experimental Value	Theoretical Value
T$_r[ms]$	200	79.9	60 ± 0.2
T$_s[ms]$	500	89.0	154.3 ± 61.9
P.O. [%]	10	2.13	8.52 ± 0.37

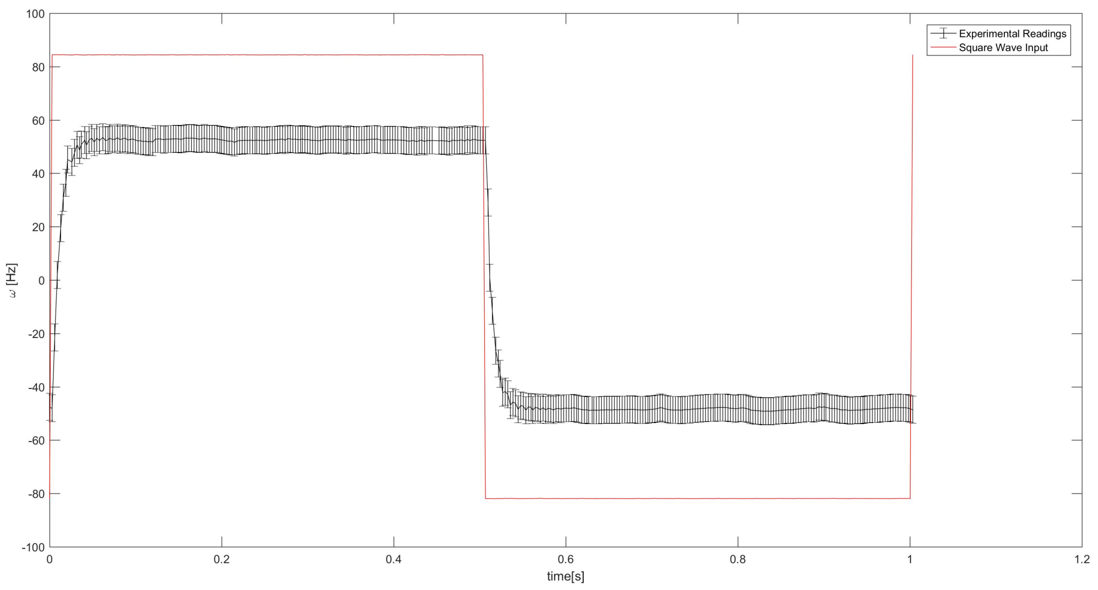


Figure 3 Experimental P Controller Input and Output Speed Versus Time

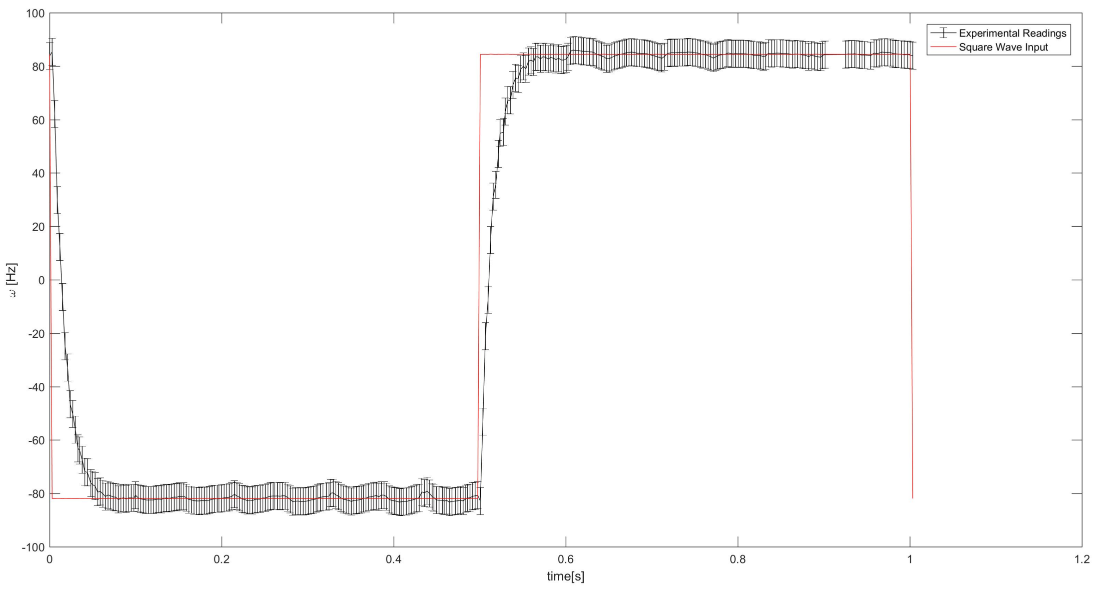


Figure 4 Experimental PI Controller Input and Output Speed Versus Time

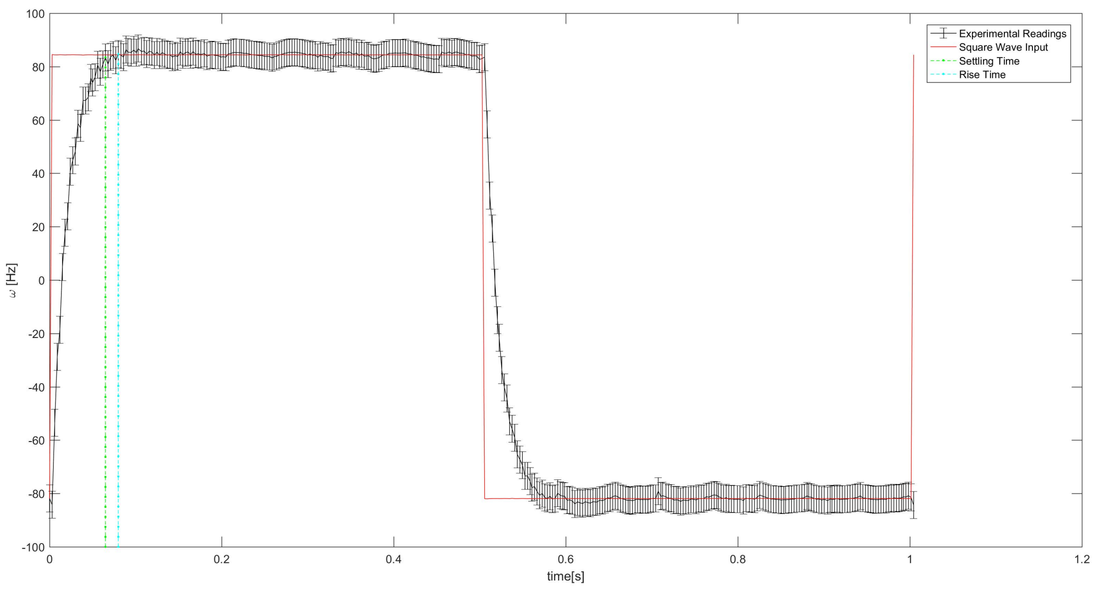


Figure 5 Experimental PID Controller Input and Output Speed Versus Time with Settling Time and Rise Time

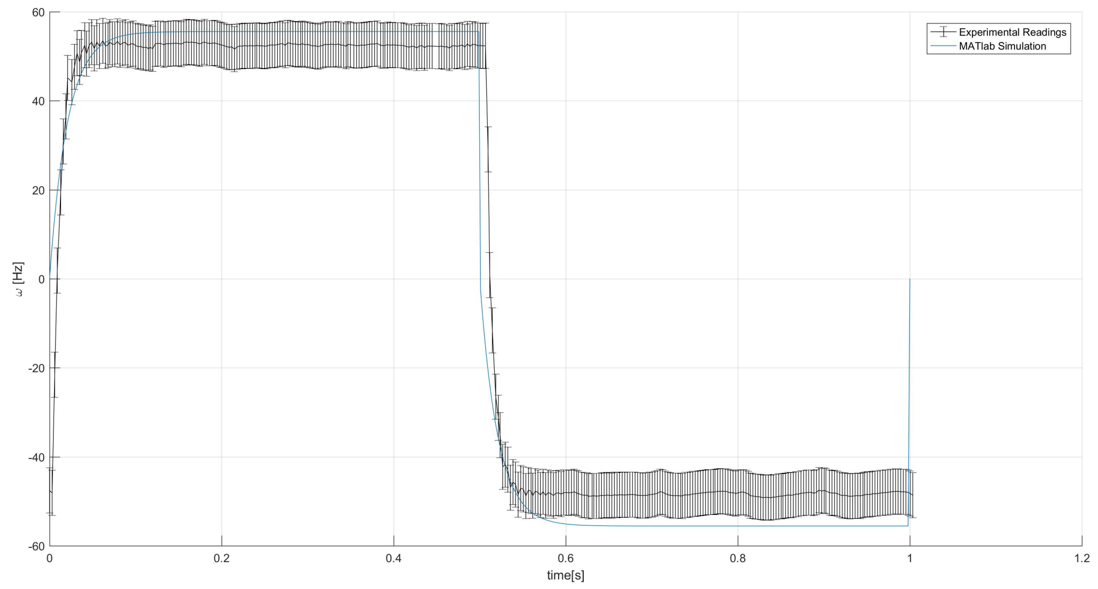


Figure 6 MATLAB Simulated and Experimental P Controller Input and Output Speed Versus Time

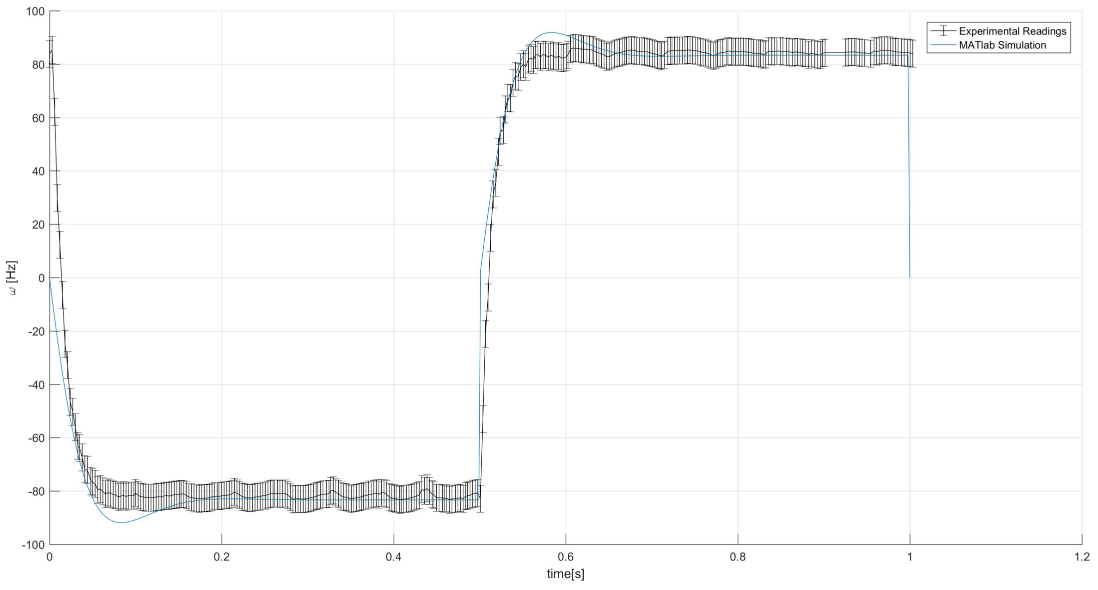


Figure 7 MATLAB Simulated and Experimental PI Controller Input and Output Speed Versus Time

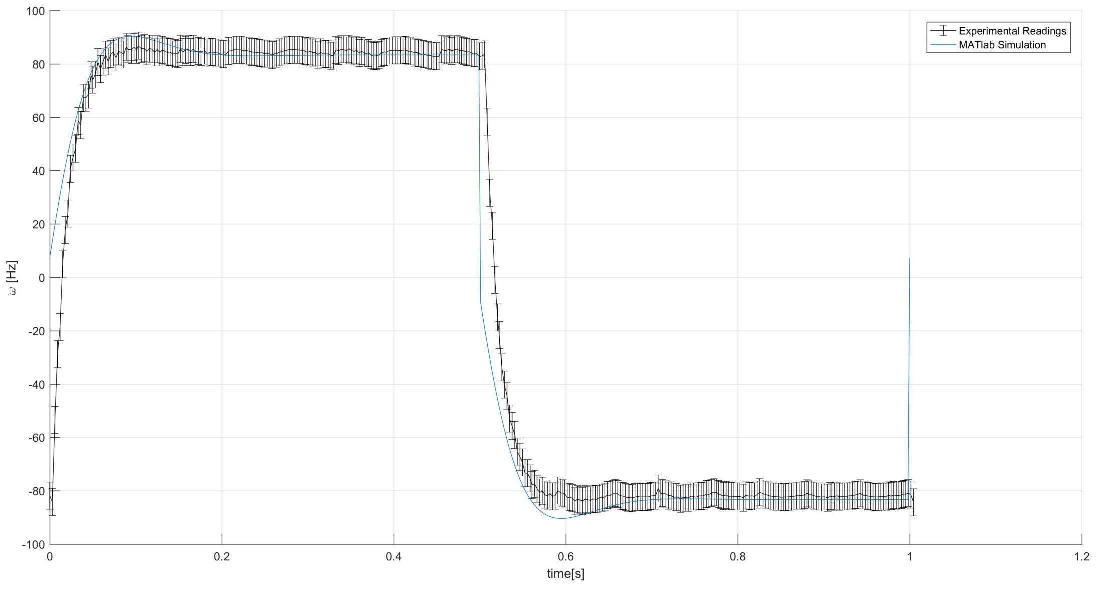


Figure 8 MATLAB Simulated and Experimental PID Controller Input and Output Speed

Versus Time

V Error Analysis

The experimental data gathered from this experiment was generated by a signal from a waveform generator used to operate an operational amplifier connected to the motor, and a DC power supply used to create the current required by the motor. While the voltage from the power supply is stepped by the circuit to match the signal of the waveform generator, errors in both are considered to have effect on the input voltage of the motor and thereby the experimental data. According to the datasheet for the waveform generator the system has a 4-digit resolution, thus an incremental uncertainty of $0.0001V$ should be applied to the waveform generator signal, ΔV_{wave-i} [5]. Additionally, the same datasheet points to a 2 mV accuracy of the generator as the manufacturer's error, so ΔV_{wave-m} of $0.002V$ is included in the calculation. Next, the power supply datasheet indicates a meter resolution of 10 mV, so an incremental error from the power supply is then applied such that ΔV_{pow-i} equals $0.01V$ [6]. Lastly, the power supply also has a manufacturer accuracy of 0.5%, once again a manufacturer error is considered such that ΔV_{pow-m} is equal to $0.005 * Vpow$ which is numerically $0.06V$, where V_{pow} is the ± 12 volts supplied by the power supply. Both of these voltage sources are connected directly in some way to the motor's power input. The waveform generator is connected through the DAQ system to the operational amplifier's positive input, which is replicated at the negative input connected to the motor. The power supply is connected to the motor when the transistor is actively drawing current from the generator to the motor. Therefore, each error is considered with equal weight in the calculation of input voltage error to the motor. As such, the input voltage error, ΔV, was calculated as follows:

$$\Delta V = \sqrt{\Delta V_{wave-i}^2 + \Delta V_{wave-m}^2 + \Delta V_{pow-i}^2 + \Delta V_{pow-m}^2}. \qquad (30)$$

ΔV can then be solved to be $\pm 0.0609\,V$. This voltage uncertainty will linearly effect the output of the motor. The uncertainty this creates in the motors output, $\Delta\omega_{theoretical}$, is calculated as follows:

$$\Delta\omega_{theoretical} = \frac{\Delta V * 10^3}{12} RPM. \qquad (31)$$

$\Delta\omega$ is calculated to be $\pm 5.07\,RPM$. This error is to be considered in the experimental reading of all three controller configurations. $\Delta\omega$ is represented in the error bars of each graph containing experimental data (Figures 3-9).

In order to assess the uncertainties of the theoretical error values described by the MATLAB simulation the mathematical elements of the time output function must be identified,

$$\omega(t) = 1/Kt - JR_aT_ie^{-t\frac{K_bK_mT_i+K_cK_mK_tT_i}{2JR_aT_i+2K_cK_mK_tT_dT_i}} *$$

$$(\cos(t * \frac{\sqrt{(K_b^2K_m^2T_i/4)-K_c^2K_m^2K_t^2T_d+(K_c^2K_m^2K_t^2T_i/4)+(K_bK_cK_m^2K_tT_i/2)-JK_cK_mK_tR_a}}{T_i^{1/2}(JR_a+K_cK_mK_tT_d)}) -$$

$$T_i^{1/2}\sin(t\frac{\sqrt{(K_b^2K_m^2T_i/4)-K_c^2K_m^2K_t^2T_d+(K_c^2K_m^2K_t^2T_i/4)+(K_bK_cK_m^2K_tT_i/2)-JK_cK_mK_tR_a}}{T_i^{1/2}(JR_a+K_cK_mK_tT_d)}) *$$

$$\frac{(\frac{K_bK_mT_i+K_cK_mK_tT_i}{2JR_aT_i+2K_cK_mK_tT_dT_i}-\frac{K_b*K_m}{J*R_a})(JR_a+K_cK_mK_tT_d)}{\sqrt{(K_b^2K_m^2T_i/4)-K_c^2K_m^2K_t^2T_d+(K_c^2K_m^2K_t^2T_i/4)+(K_bK_cK_m^2K_tT_i/2)-JK_cK_mK_tR_a}})/(K_t(JR_aT_i+K_cK_mK_tT_dT_i))$$

.(32)

The natural frequency, ω_n, and the damping ratio, ξ, can be identified by observing the above equation in the following format:

$$1 - e^{-\varepsilon\omega_n t}\left(\cos\omega_n\sqrt{1-\xi^2} + + \frac{\xi}{\sqrt{1-\xi}}\sin\omega_n\sqrt{1-\xi^2}\right). \qquad (33)$$

Thus, the natural frequency and the damping ratio can be identified as the following:

$$\omega_n = \frac{1}{T_i^{1/2}(JR_a + K_cK_mK_tT_d)} \tag{34}$$

and

$$\xi = T_i^{1/2}\frac{K_bK_m + K_cK_mK_t}{4}. \tag{35}$$

These values solve to be $\omega_n = 1.0213x10^3 \pm \Delta\omega_n$ Hz and $\xi = 0.0107 \pm \Delta\xi$ The error in ω_n and ξ can be propagated with the errors and partial derivatives of K_b, K_m, and R_a. The error ΔK_b is equal to 0.1, while K_b is equal to $1.2\frac{V}{KRPM}$ and $7.2X10^{-2}\frac{V}{Hz}$. The error ΔK_m is equal to 0.1, while K_m is equal to $1.62\frac{oz-in}{A}$ and $1.144X10^{-}2\frac{Nm}{A}$. Lastly, the error ΔR_a is equal to 0.15, and R_a is equal to 1.725Ω. The partial derivatives of ω_n with respect to K_m and R_a are listed as follows:

$$\frac{\partial\omega_n}{\partial K_m} = -\frac{K_cT_dK_t}{\sqrt{T_i}(R_aJ + K_cT_dK_mK_t)^2} \tag{36}$$

and

$$\frac{\partial\omega_n}{\partial R_a} = -\frac{J}{\sqrt{T_i}(R_aJ + K_cT_dK_mK_t)^2}. \tag{37}$$

By obtaining the root sum square of the product of each partial derivative and each of there respective errors the following can obtained for the total error in ω_n:

$$\Delta\omega_n = \sqrt{(-\frac{K_cT_dK_t\Delta K_m}{\sqrt{T_i}(R_aJ + K_cT_dK_mK_t)^2})^2 + (-\frac{J\Delta R_a}{\sqrt{T_i}(R_aJ + K_cT_dK_mK_t)^2})^2}. \tag{38}$$

Numerically this equates as $\Delta\omega_n$ is equal to $\pm 140.0867\,Hz$.

Similarly, the partial derivatives of ξ in terms of K_b and K_m are:

$$\frac{\partial\xi}{\partial K_b} = T_i^{1/2}\frac{K_m}{4}, \tag{39}$$

$$\frac{\partial \xi}{\partial K_m} = T_i^{1/2} \frac{K_b + K_c K_t}{4} \tag{40}$$

By obtaining the root sum square of the product of each partial derivative and each of their respective errors, total error in ξ can be obtained:

$$\Delta \xi = \sqrt{(T_i^{1/2} \frac{K_m \Delta K_b}{4})^2 + (T_i^{1/2} \frac{(K_b + K_c K_t) \Delta K_m}{4})^2}. \tag{41}$$

Numerically this equates as $\Delta \xi$ is equal to ± 0.0011. Rise time, T_r, is defined as:

$$T_r = \frac{\pi - \cos^{-1}(\xi)}{\omega_n \sqrt{1 - \xi^2}}. \tag{42}$$

The partial derivative of the rise time is in respect to ω_n and ξ is:

$$\frac{\partial T_r}{\partial \omega_n} = -\frac{\pi - \cos^{-1}(\xi)}{\omega_n^2 \sqrt{1 - \xi^2}} \tag{43}$$

and

$$\frac{\partial T_r}{\partial \xi} = \frac{1}{\omega_n (1 - \xi^2)} + \frac{\xi (\pi - \cos^{-1}(\xi))}{\omega_n (1 - \xi^2)^{3/2}}, \tag{44}$$

respectively. By obtaining the root sum square of the product between each partial derivative and each of there respective errors, the total error in T_r can be obtained:

$$\Delta T_r = \sqrt{(-\frac{\pi - \cos^{-1}(\xi)}{\omega_n^2 \sqrt{1 - \xi^2}} \Delta \omega_n)^2 + ((\frac{1}{\omega_n (1 - \xi^2)} + \frac{\xi (\pi - \cos^{-1}(\xi))}{\omega_n (1 - \xi^2)^{3/2}}) \Delta \xi)^2}. \tag{45}$$

Numerically, the error in the rise time equates to $\pm 2.124 X 10^{-4}$ seconds.

The percent overshoot formula, $P.O.$, is defined as follows:

$$P.O. = 100 e^{\frac{-\xi \pi}{\sqrt{1 - \xi^2}}}. \tag{46}$$

The only component with a known error in this formula is ξ. Therefore, the partial derivative in terms of ξ must be obtained,

$$\frac{\partial P.O.}{\partial \xi} = \frac{100\pi e^{\frac{\pi\xi}{\sqrt{1-\xi^2}}}}{(1-\xi^2)^{3/2}}. \tag{47}$$

By multiplying the error in the damping ratio, $\Delta\xi$ with the partial derivative of $P.O.$ in terms of ξ the error in the theoretical percent overshoot can be obtained.

$$\Delta P.O. = \frac{100\pi e^{\frac{\pi\xi}{\sqrt{1-\xi^2}}}}{(1-\xi^2)^{3/2}}\Delta\xi \tag{48}$$

The error, $\Delta P.O.$, solves to be $\pm 0.3666\%$. Next, the settling time is the time when the value settles to 2% of the expected value and it is represented as follows:

$$T_s = \frac{ln(2/100)}{\omega_n \xi}. \tag{49}$$

As done previously the partial derivatives are taken for ω_n and ξ respectively:

$$\frac{\partial T_s}{\partial \omega_n} = -\frac{ln(2/100)}{\omega_n^2 \xi} \tag{50}$$

and

$$\frac{\partial T_s}{\partial \xi} = -\frac{ln(2/100)}{\omega_n \xi^2} \tag{51}$$

The root sum square of the product between each error and its respective partial derivative is then taken,

$$\Delta T_s = \sqrt{(-\frac{\omega_n ln(2/100)}{\omega_n^2 \xi})^2 + (-\frac{\xi ln(2/100)}{\omega_n \xi^2})^2}. \tag{52}$$

Numerically, the error in the settling time solves to be ± 0.0619 seconds. This method of numerical error analysis is crucial for determining the validity of the results by establishing uncertainty ranges which the values can fall within. Otherwise, there is no way to tell if a discrepancy between theoretical and experimental values is valid or not.

VI Discussion

Both the PI and PID controller outputs can be graphically observed to contain the square waveform within their uncertainties for the majority of the observed data. While the P controller, understandably, does not produce results that meet the criteria of the design. The PID controller specifically meets the specification of the design. The rise time was graphically determined to be 79.9 ms, well below the 200 ms specification. Similarly, the percent overshoot was calculated to be 2.13%, well within the 10% limit set by the specifications. Lastly, the settling time was 89.0 ms which is just after the rise time. The rise time was within the specification of 500 ms, which consisted of the entire half of the waveform. All of the data described can be observed in the experimental PID controller output graph (Figure 5). The theoretical values for these specifications are observed graphically from the simulation. The theoretical rise time is 60 ± 0.2 ms, this value is outside of uncertainty of the observed 79.9 ms. The theoretical peak over shoot is $8.52\% \pm 0.37\%$ once again, outside of the uncertainty for the experimental 2.13%. Lastly, the theoretical value for settling time is $154.3 \pm 61.9 ms$, once again outside of the uncertainty for the experimental value of $89.0\, ms$. These theoretical values seam to describe a function with a much more extreme rise that takes more time to settle after than the experimental signal. In order to further investigate the cause of these discrepancies the experimental functions are compared graphically to there simulated counterparts (Figures 6-8).

The MATLAB simulation contained several data points that were outside of uncertainty of the experimental results for each controller. The most grievous case observed is that of the simulated P controller (Figure 6). The entirety of the second half of the waveform is outside the uncertainty of the experimental data. This is abnormal since if there was an error in amplitude one would expect both halves of the waveform to appear abnormal. Since only one side is outside of uncertainty it is possible that there is some offset in the data caused by an inaccurate ground. This would also explain why the peak on the lower half of the PID controller simulation is outside of uncertainty but the upper peak is not. If the ground is not at a perfect zero voltage, it should be expected that the input voltage to the motor is off by a magnitude of the ground's true value. This would result in a similar offset in the motor output. The RPM equivalent of the waveform generator output is observe to find the value at which it settles at on the positive and negative half of the wave. As predicted, there was a discrepancy in magnitude of the positive and negative settling values for the waveform generator. The positive ends settles at 84.44 RPM while the negative end settles around -81.90 RPM. The offset, ω_{off}, is calculated as follows:

$$\omega_{off} = \frac{84.44 - 81.90}{2} = 1.25\, RPM \tag{53}$$

Such a ground offset would not fully account for the PID controller's discrepancy from theoretical values at its peaks. The feature with the greatest effect on the output peak would be the motor's back EMF constant, K_b. If there was some damage to the solenoid driving the electric motor, there could be an abnormally higher back EMF constant at play.

The ω_{off} was taken into account for the simulation by offsetting the simulated data by +1.25V. Additionally the back EMF constant was increased until the peak of the simulated data replicated the the central experimental trend. The final adjusted value of the back EMF constant was $19\frac{V}{KRPM}$. With the offset and back EMF adjusted the resulting simulation not only settled at the same value and the experimental data thanks to the ground offset, the adjusted back EMF resulted in identical peaks between the two sets. It can be noted to obtain the perfect peak match the back EMF was increased significantly in the simulation, it can be increased to a much smaller degree to be with the uncertainty of the error analysis. Therefore, it is more likely that the true back EMF value was $13.2\frac{V}{KRPM}$ at the upper end of its 10% uncertainty.

It should also be observed that in all three simulations, the experimental values failed to replicate the experimental values between steps from positive and negative intervals. this is due to the fact that the simulation is modeled as a piece-wise function. Meaning the step from positive to negative is not dictated by a change of input, but by a change of sign in the governing mathematical formula. As such, these simulations should not be considered for accurate estimation of motor speed between steps.

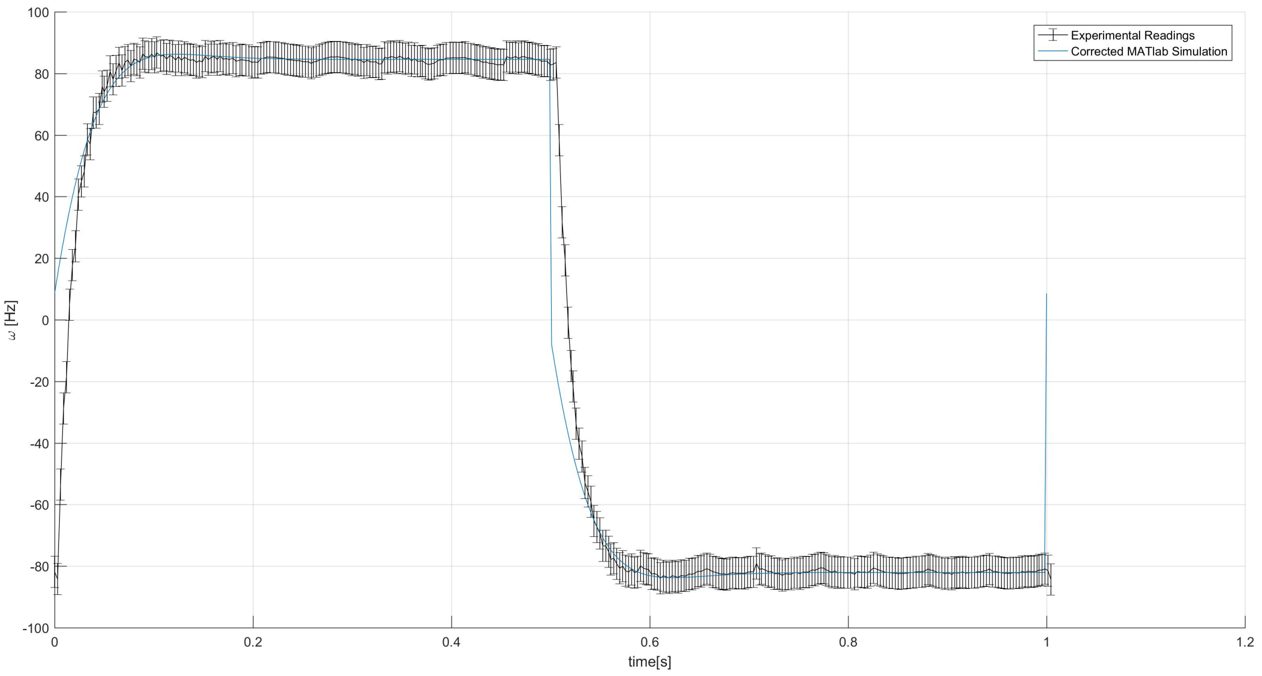


Figure 9 MATLAB Simulated and Experimental PID Controller Input and Output Speed Versus Time with Corrected Discrepancies

VII Conclusions

- The experimental results showed that a PID turntable speed control system can be created with accurate results.
- The relevant values were found to be the control variables of the PID, PI, and P controllers (Table 1) and the settling time, percent overshoot, and rise time of the PID controller (Table 3).
- Both the simulated and actual PID controllers met the specifications of the design.
- The experimental results did not agree with the theoretical results for the P, PI, or PID controllers for all data points.
- The discrepancy is attributed to an inaccuracy in the ground of the power supply, an abnormally high back EMF constant, and the theoretical simulation not accounting for the motor speed between steps.
- In future investigations, the level of difficulty in translating the transfer function to a function of output over time using inverse Laplace transform should be considered. The inverse Laplace of the transfer function was not done by hand because of the difficulty, which made error analysis even more difficult. Therefore, more guidance should have been provided in terms of this section to enable a better mathematical analysis on the error for the desired values.

VIII References

[1] Machtay, Noah Donald, Ph.D., P.E. MEC 411 Control System Analysis and Design. Fall 2019. SUNY Stony Brook University Department of Mechanical Engineering College of Engineering and Applied Sciences

[2] Machtay, Noah Donald, Ph.D., P.E. “MEC 411 Digital PID Speed Control of a Turntable.” SUNY Stony Brook University Department of Mechanical Engineering College of Engineering and Applied Sciences

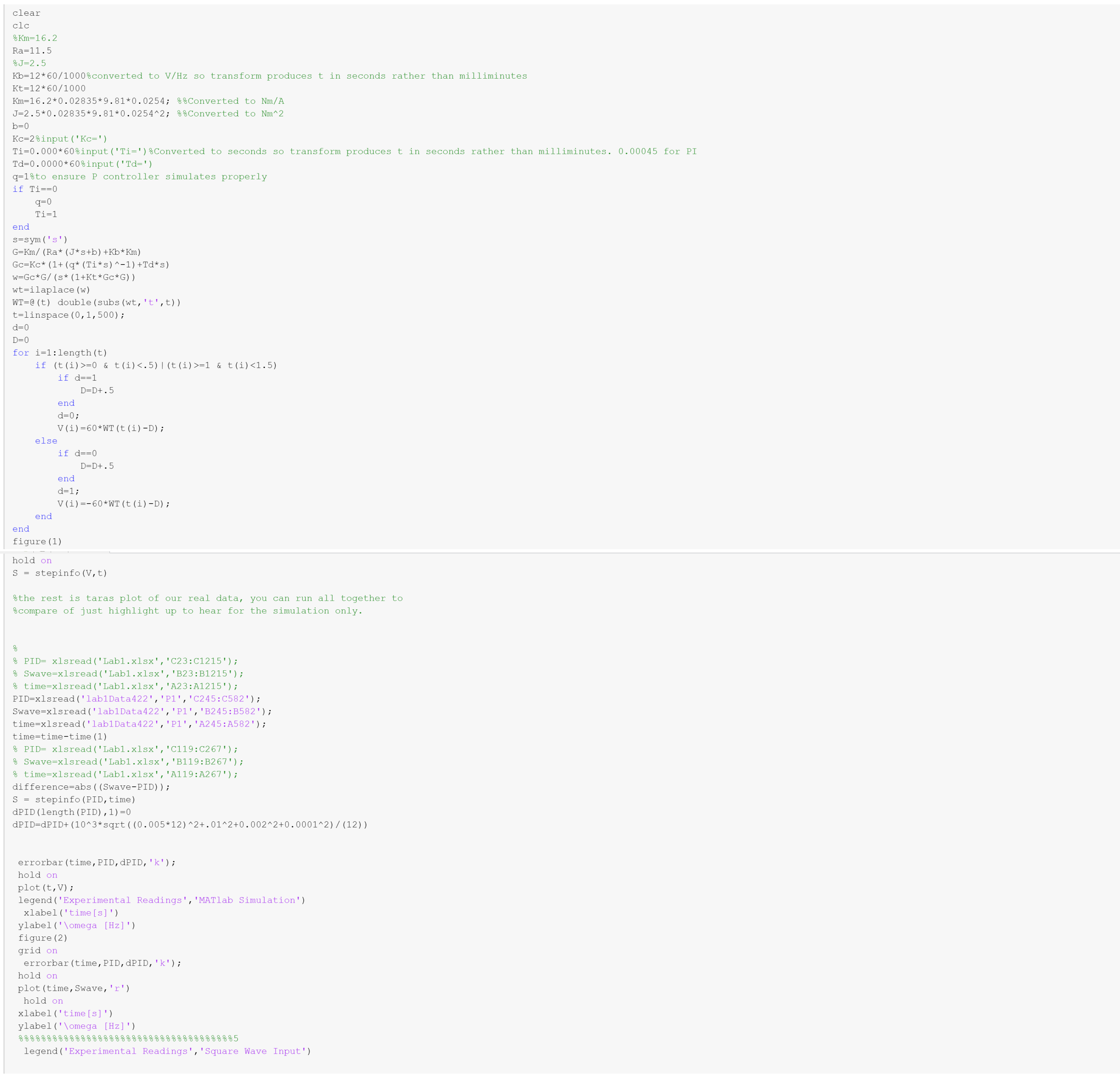
[3] “What Is a Tachometer?” *AZoSensors.com*, 8 Aug. 2019, www.azosensors.com/article.aspx?ArticleID=310.

[4] Gene F. Franklin,J. David Powell, and Abbas Emami-Naeini, “Feedback Control of Dynamic Systems”, 6th ed., Prentice Hall, 2010.

[5] “Keysight 33220A 20 MHz Function/Arbitrary Waveform Generator Data Sheet,” *Keysight Technologies*, http://www.testequipmenthq.com/datasheets/Agilent-E3630A-Datasheet.pdf.

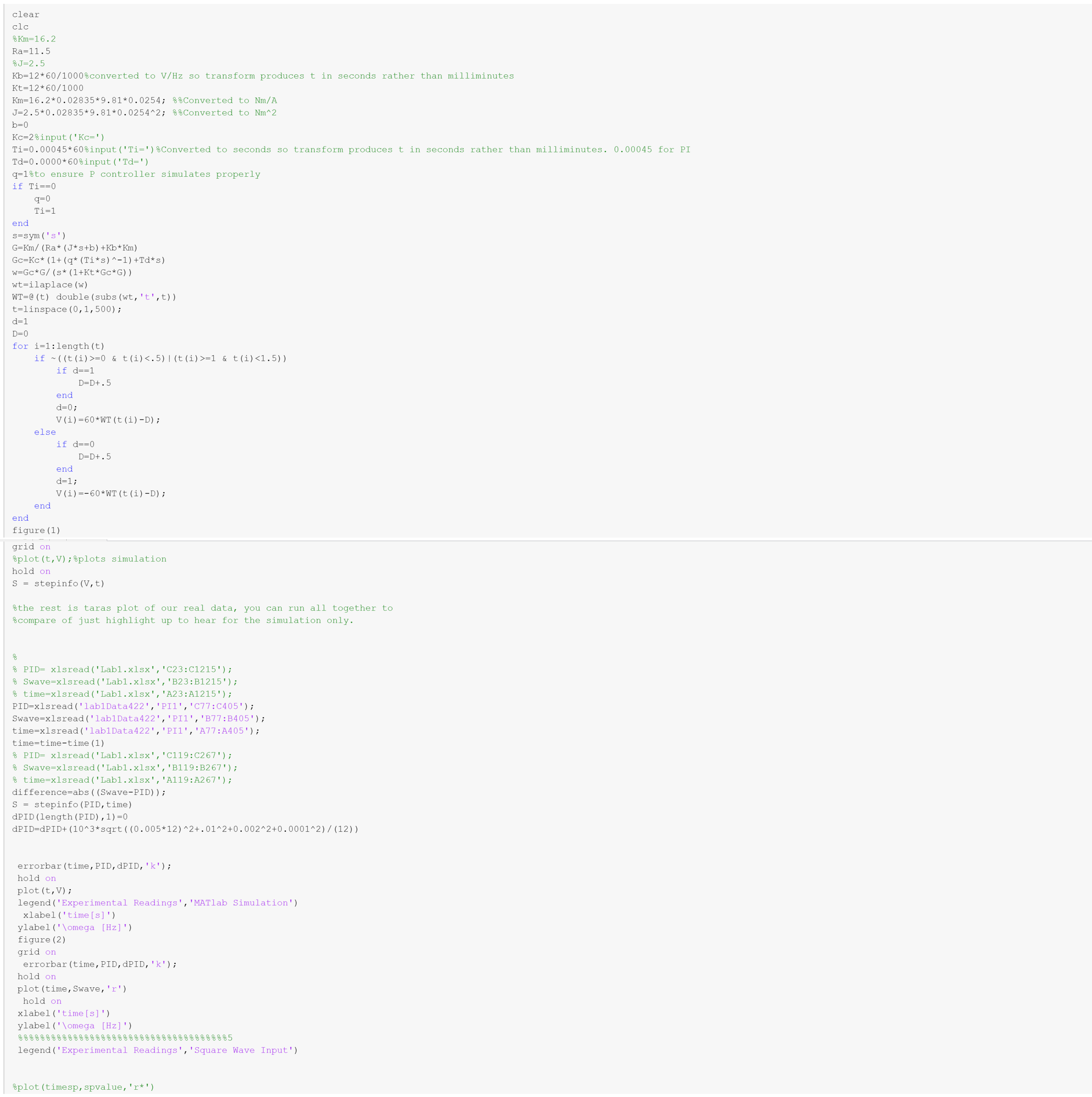
[6] “Agilent E36XXA Series Non-Programmable DC Power Supplies Data Sheet,” *Agilent Technologies*, https://literature.cdn.keysight.com/litweb/pdf/5988-8544EN.pdf?id=187648.

IX Appendix



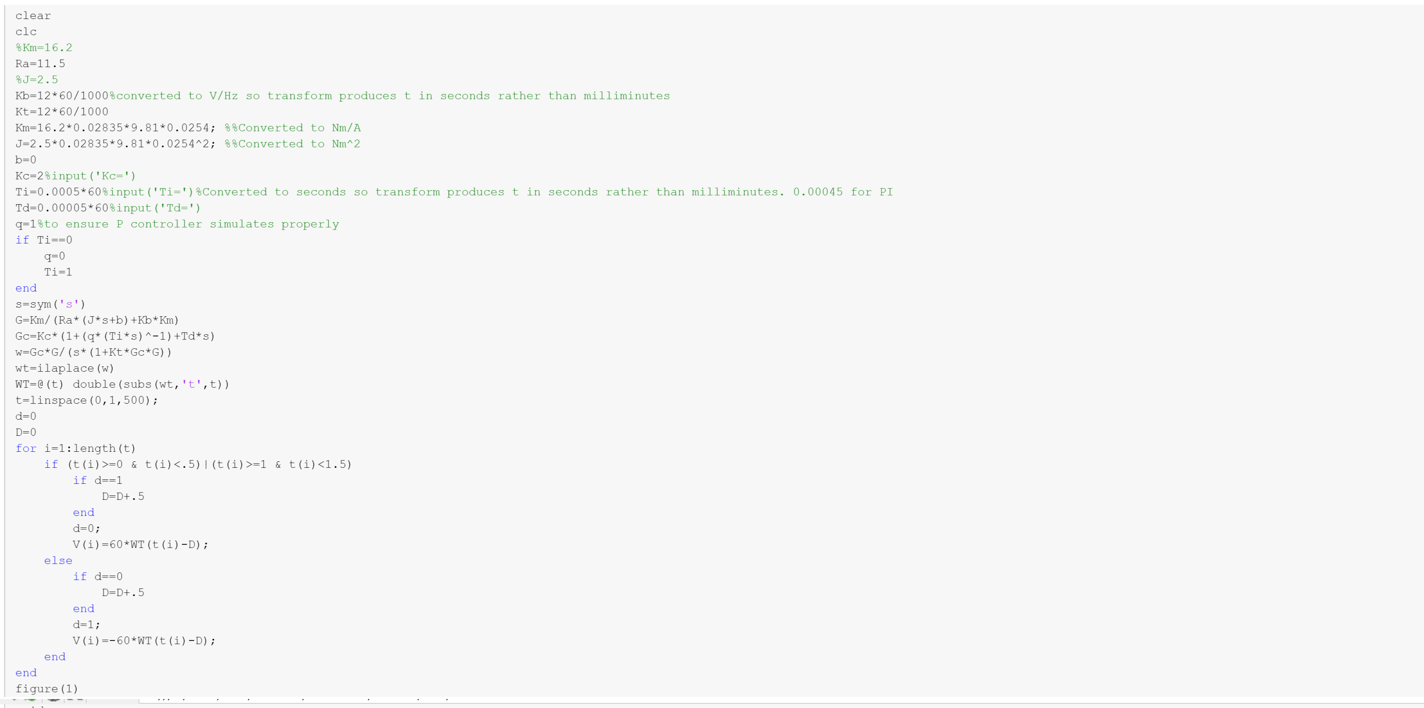
```
clear
clc
%Km=16.2
Ra=11.5
%J=2.5
Kb=12*60/1000%converted to V/Hz so transform produces t in seconds rather than milliminutes
Kt=12*60/1000
Km=16.2*0.02835*9.81*0.0254; %%Converted to Nm/A
J=2.5*0.02835*9.81*0.0254^2; %%Converted to Nm^2
b=0
Kc=2%input('Kc=')
Ti=0.000*60%input('Ti=')%Converted to seconds so transform produces t in seconds rather than milliminutes. 0.00045 for PI
Td=0.0000*60%input('Td=')
q=1%to ensure P controller simulates properly
if Ti==0
    q=0
    Ti=1
end
s=sym('s')
G=Km/(Ra*(J*s+b)+Kb*Km)
Gc=Kc*(1+(q*(Ti*s)^-1)+Td*s)
w=Gc*G/(s*(1+Kt*Gc*G))
wt=ilaplace(w)
WT=@(t) double(subs(wt,'t',t))
t=linspace(0,1,500);
d=0
D=0
for i=1:length(t)
    if (t(i)>=0 & t(i)<.5)|(t(i)>=1 & t(i)<1.5)
        if d==1
            D=D+.5
        end
        d=0;
        V(i)=60*WT(t(i)-D);
    else
        if d==0
            D=D+.5
        end
        d=1;
        V(i)=-60*WT(t(i)-D);
    end
end
figure(1)
hold on
S = stepinfo(V,t)

%the rest is taras plot of our real data, you can run all together to
%compare of just highlight up to hear for the simulation only.


%
% PID= xlsread('Lab1.xlsx','C23:C1215');
% Swave=xlsread('Lab1.xlsx','B23:B1215');
% time=xlsread('Lab1.xlsx','A23:A1215');
PID=xlsread('lab1Data422','P1','C245:C582');
Swave=xlsread('lab1Data422','P1','B245:B582');
time=xlsread('lab1Data422','P1','A245:A582');
time=time-time(1)
% PID= xlsread('Lab1.xlsx','C119:C267');
% Swave=xlsread('Lab1.xlsx','B119:B267');
% time=xlsread('Lab1.xlsx','A119:A267');
difference=abs((Swave-PID));
S = stepinfo(PID,time)
dPID(length(PID),1)=0
dPID=dPID+(10^3*sqrt((0.005*12)^2+.01^2+0.002^2+0.0001^2)/(12))


 errorbar(time,PID,dPID,'k');
 hold on
 plot(t,V);
 legend('Experimental Readings','MATlab Simulation')
  xlabel('time[s]')
 ylabel('\omega [Hz]')
 figure(2)
 grid on
  errorbar(time,PID,dPID,'k');
 hold on
 plot(time,Swave,'r')
  hold on
 xlabel('time[s]')
 ylabel('\omega [Hz]')
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5
  legend('Experimental Readings','Square Wave Input')
```

Figure A1 P Controller MATLAB Code

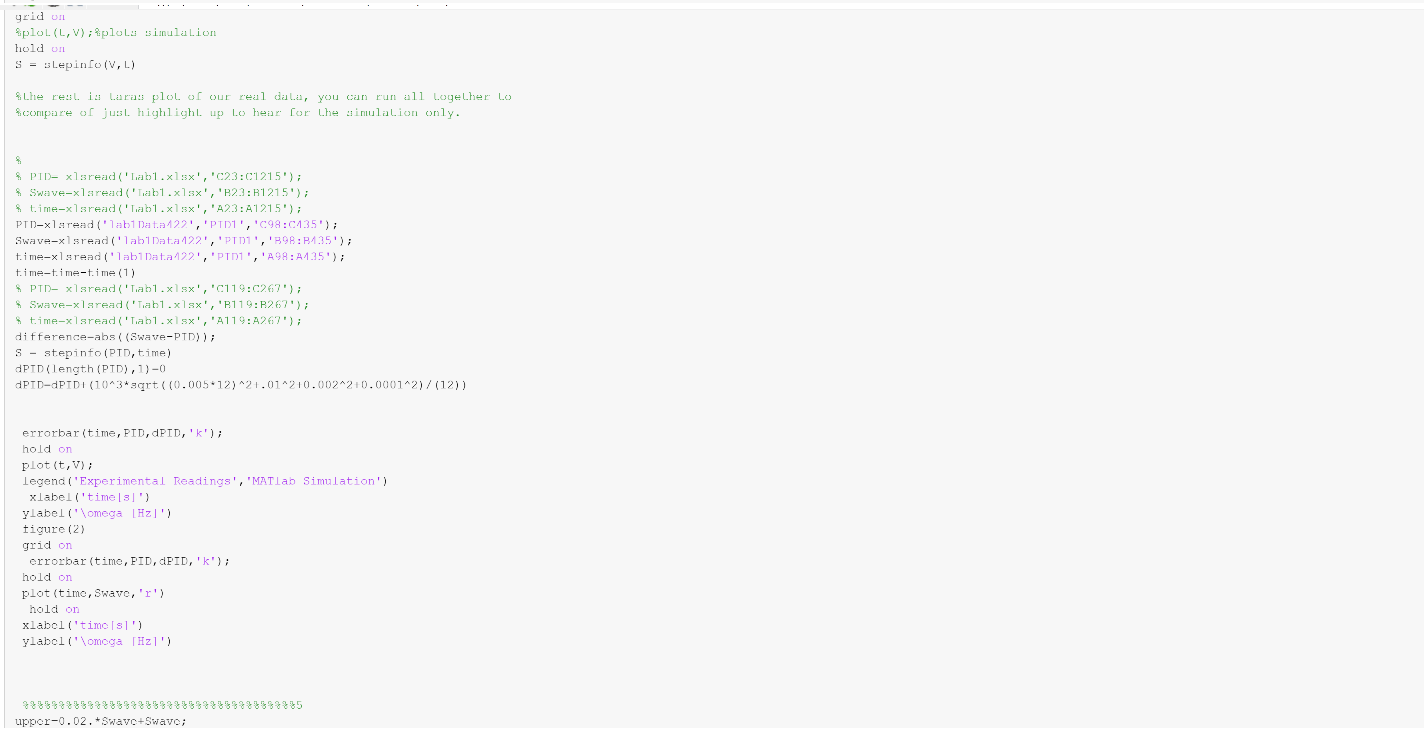


```
clear
clc
%Km=16.2
Ra=11.5
%J=2.5
Kb=12*60/1000%converted to V/Hz so transform produces t in seconds rather than milliminutes
Kt=12*60/1000
Km=16.2*0.02835*9.81*0.0254; %%Converted to Nm/A
J=2.5*0.02835*9.81*0.0254^2; %%Converted to Nm^2
b=0
Kc=2%input('Kc=')
Ti=0.00045*60%input('Ti=')%Converted to seconds so transform produces t in seconds rather than milliminutes. 0.00045 for PI
Td=0.0000*60%input('Td=')
q=1%to ensure P controller simulates properly
if Ti==0
    q=0
    Ti=1
end
s=sym('s')
G=Km/(Ra*(J*s+b)+Kb*Km)
Gc=Kc*(1+(q*(Ti*s)^-1)+Td*s)
w=Gc*G/(s*(1+Kt*Gc*G))
wt=ilaplace(w)
WT=@(t) double(subs(wt,'t',t))
t=linspace(0,1,500);
d=1
D=0
for i=1:length(t)
    if ~((t(i)>=0 & t(i)<.5)|(t(i)>=1 & t(i)<1.5))
        if d==1
            D=D+.5
        end
        d=0;
        V(i)=60*WT(t(i)-D);
    else
        if d==0
            D=D+.5
        end
        d=1;
        V(i)=-60*WT(t(i)-D);
    end
end
figure(1)
grid on
%plot(t,V);%plots simulation
hold on
S = stepinfo(V,t)

%the rest is taras plot of our real data, you can run all together to
%compare of just highlight up to hear for the simulation only.


%
% PID= xlsread('Lab1.xlsx','C23:C1215');
% Swave=xlsread('Lab1.xlsx','B23:B1215');
% time=xlsread('Lab1.xlsx','A23:A1215');
PID=xlsread('lab1Data422','PI1','C77:C405');
Swave=xlsread('lab1Data422','PI1','B77:B405');
time=xlsread('lab1Data422','PI1','A77:A405');
time=time-time(1)
% PID= xlsread('Lab1.xlsx','C119:C267');
% Swave=xlsread('Lab1.xlsx','B119:B267');
% time=xlsread('Lab1.xlsx','A119:A267');
difference=abs((Swave-PID));
S = stepinfo(PID,time)
dPID(length(PID),1)=0
dPID=dPID+(10^3*sqrt((0.005*12)^2+.01^2+0.002^2+0.0001^2)/(12))


 errorbar(time,PID,dPID,'k');
 hold on
 plot(t,V);
 legend('Experimental Readings','MATlab Simulation')
  xlabel('time[s]')
 ylabel('\omega [Hz]')
 figure(2)
 grid on
  errorbar(time,PID,dPID,'k');
 hold on
 plot(time,Swave,'r')
  hold on
 xlabel('time[s]')
 ylabel('\omega [Hz]')
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5
 legend('Experimental Readings','Square Wave Input')


%plot(timesp,spvalue,'r*')
```

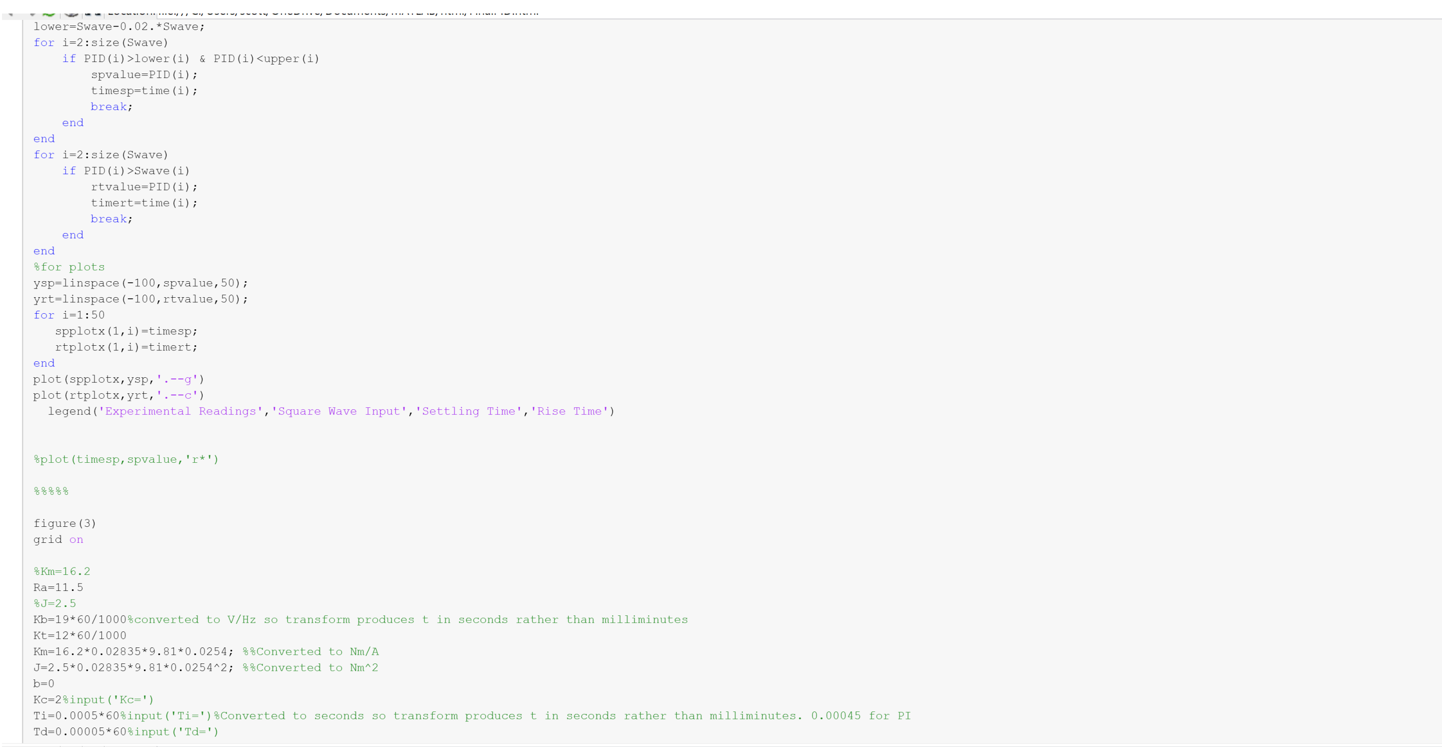
Figure A2 PI Controller MATLAB Code



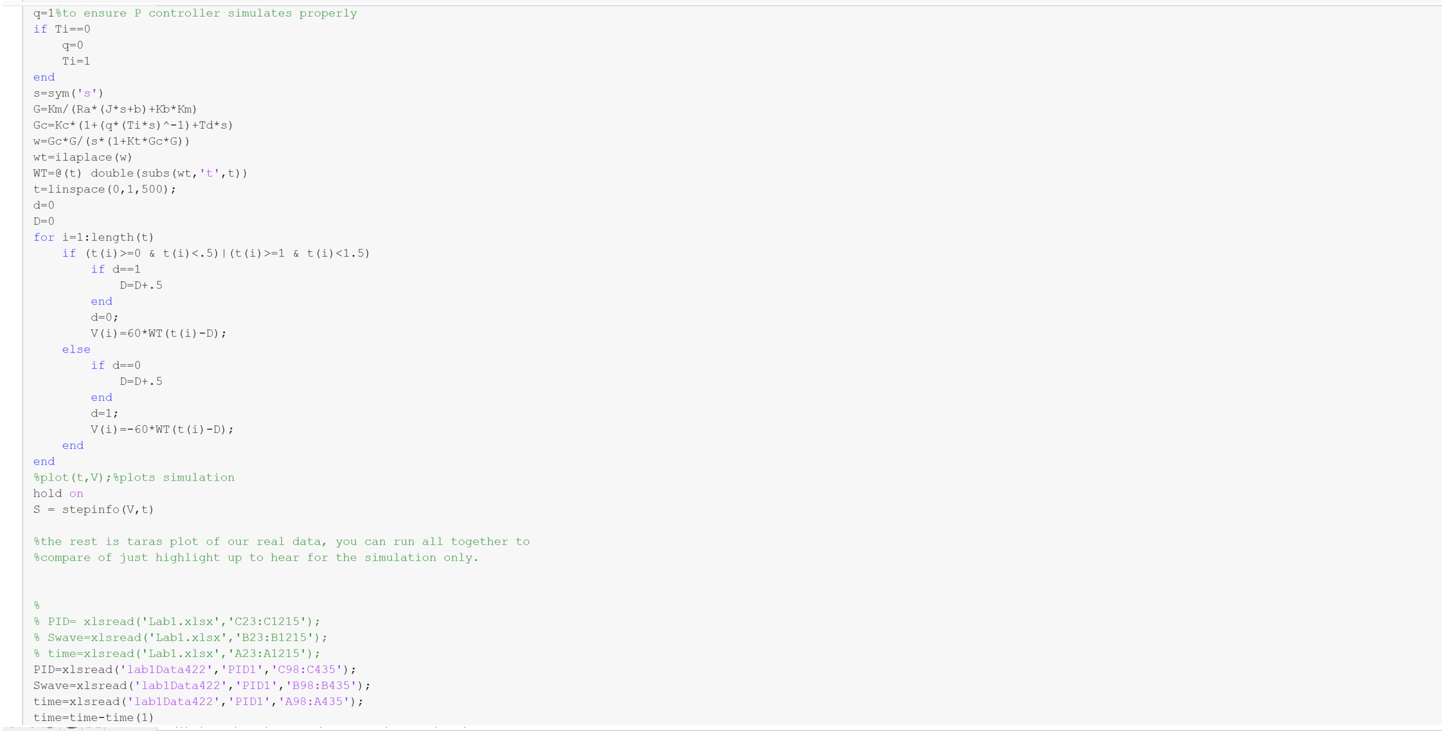
```
clear
clc
%Km=16.2
Ra=11.5
%J=2.5
Kb=12*60/1000%converted to V/Hz so transform produces t in seconds rather than milliminutes
Kt=12*60/1000
Km=16.2*0.02835*9.81*0.0254; %%Converted to Nm/A
J=2.5*0.02835*9.81*0.0254^2; %%Converted to Nm^2
b=0
Kc=2%input('Kc=')
Ti=0.0005*60%input('Ti=')%Converted to seconds so transform produces t in seconds rather than milliminutes. 0.00045 for PI
Td=0.00005*60%input('Td=')
q=1%to ensure P controller simulates properly
if Ti==0
    q=0
    Ti=1
end
s=sym('s')
G=Km/(Ra*(J*s+b)+Kb*Km)
Gc=Kc*(1+(q*(Ti*s)^-1)+Td*s)
w=Gc*G/(s*(1+Kt*Gc*G))
wt=ilaplace(w)
WT=@(t) double(subs(wt,'t',t))
t=linspace(0,1,500);
d=0
D=0
for i=1:length(t)
    if (t(i)>=0 & t(i)<.5)|(t(i)>=1 & t(i)<1.5)
        if d==1
            D=D+.5
        end
        d=0;
        V(i)=60*WT(t(i)-D);
    else
        if d==0
            D=D+.5
        end
        d=1;
        V(i)=-60*WT(t(i)-D);
    end
end
figure(1)
```



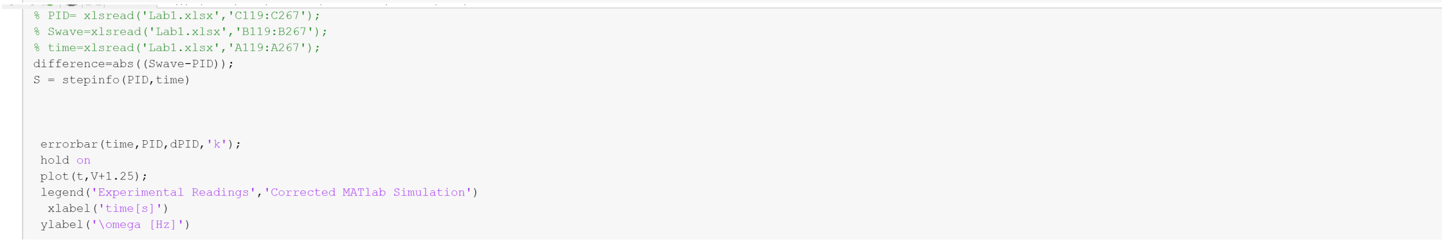
```
grid on
%plot(t,V);%plots simulation
hold on
S = stepinfo(V,t)

%the rest is taras plot of our real data, you can run all together to
%compare of just highlight up to hear for the simulation only.


%
% PID= xlsread('Lab1.xlsx','C23:C1215');
% Swave=xlsread('Lab1.xlsx','B23:B1215');
% time=xlsread('Lab1.xlsx','A23:A1215');
PID=xlsread('lab1Data422','PID1','C98:C435');
Swave=xlsread('lab1Data422','PID1','B98:B435');
time=xlsread('lab1Data422','PID1','A98:A435');
time=time-time(1)
% PID= xlsread('Lab1.xlsx','C119:C267');
% Swave=xlsread('Lab1.xlsx','B119:B267');
% time=xlsread('Lab1.xlsx','A119:A267');
difference=abs((Swave-PID));
S = stepinfo(PID,time)
dPID(length(PID),1)=0
dPID=dPID+(10^3*sqrt((0.005*12)^2+.01^2+0.002^2+0.0001^2)/(12))


 errorbar(time,PID,dPID,'k');
 hold on
 plot(t,V);
 legend('Experimental Readings','MATlab Simulation')
  xlabel('time[s]')
 ylabel('\omega [Hz]')
 figure(2)
 grid on
  errorbar(time,PID,dPID,'k');
 hold on
 plot(time,Swave,'r')
  hold on
 xlabel('time[s]')
 ylabel('\omega [Hz]')



 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5
upper=0.02.*Swave+Swave;
```



```
lower=Swave-0.02.*Swave;
for i=2:size(Swave)
    if PID(i)>lower(i) & PID(i)<upper(i)
        spvalue=PID(i);
        timesp=time(i);
        break;
    end
end
for i=2:size(Swave)
    if PID(i)>Swave(i)
        rtvalue=PID(i);
        timert=time(i);
        break;
    end
end
%for plots
ysp=linspace(-100,spvalue,50);
yrt=linspace(-100,rtvalue,50);
for i=1:50
   spplotx(1,i)=timesp;
   rtplotx(1,i)=timert;
end
plot(spplotx,ysp,'.--g')
plot(rtplotx,yrt,'.--c')
  legend('Experimental Readings','Square Wave Input','Settling Time','Rise Time')


%plot(timesp,spvalue,'r*')

%%%%%

figure(3)
grid on

%Km=16.2
Ra=11.5
%J=2.5
Kb=19*60/1000%converted to V/Hz so transform produces t in seconds rather than milliminutes
Kt=12*60/1000
Km=16.2*0.02835*9.81*0.0254; %%Converted to Nm/A
J=2.5*0.02835*9.81*0.0254^2; %%Converted to Nm^2
b=0
Kc=2%input('Kc=')
Ti=0.0005*60%input('Ti=')%Converted to seconds so transform produces t in seconds rather than milliminutes. 0.00045 for PI
Td=0.00005*60%input('Td=')
```



```
q=1%to ensure P controller simulates properly
if Ti==0
    q=0
    Ti=1
end
s=sym('s')
G=Km/(Ra*(J*s+b)+Kb*Km)
Gc=Kc*(1+(q*(Ti*s)^-1)+Td*s)
w=Gc*G/(s*(1+Kt*Gc*G))
wt=ilaplace(w)
WT=@(t) double(subs(wt,'t',t))
t=linspace(0,1,500);
d=0
D=0
for i=1:length(t)
    if (t(i)>=0 & t(i)<.5)|(t(i)>=1 & t(i)<1.5)
        if d==1
            D=D+.5
        end
        d=0;
        V(i)=60*WT(t(i)-D);
    else
        if d==0
            D=D+.5
        end
        d=1;
        V(i)=-60*WT(t(i)-D);
    end
end
%plot(t,V);%plots simulation
hold on
S = stepinfo(V,t)

%the rest is taras plot of our real data, you can run all together to
%compare of just highlight up to hear for the simulation only.


%
% PID= xlsread('Lab1.xlsx','C23:C1215');
% Swave=xlsread('Lab1.xlsx','B23:B1215');
% time=xlsread('Lab1.xlsx','A23:A1215');
PID=xlsread('lab1Data422','PID1','C98:C435');
Swave=xlsread('lab1Data422','PID1','B98:B435');
time=xlsread('lab1Data422','PID1','A98:A435');
time=time-time(1)
```



```
% PID= xlsread('Lab1.xlsx','C119:C267');
% Swave=xlsread('Lab1.xlsx','B119:B267');
% time=xlsread('Lab1.xlsx','A119:A267');
difference=abs((Swave-PID));
S = stepinfo(PID,time)



 errorbar(time,PID,dPID,'k');
 hold on
 plot(t,V+1.25);
 legend('Experimental Readings','Corrected MATlab Simulation')
  xlabel('time[s]')
 ylabel('\omega [Hz]')
```

Figure A3 PID Controller MATLAB Code

Comment Summary

Page 3

1. 2 not 0

Page 6

2. This symbol means convolution in Laplace domain. So try to use (.) as simple multiplication.
3. Contradiction between Equation 9 and 10.
4. Substitute Eq 11 - 15 into Eq 10 to obtain the final expression of T(s).

Page 7

5. cite

Page 10

6. cite

Page 21

7. \times in Latex
8. citation needed

Page 23

9. use siunitx package for unit in Latex

Page 29

10. larger size or use listings package in latex