Lab 3: Digital PID Speed Control of a Turntable

Group 1

Jeremy Nielsen

Tara Boyle

Scott Miller

November 18, 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 design and subsequently implement a proportional-integral-derivative (PID) feedback loop control system to control the rotational speed of a turntable. An digital electrical circuit was constructed to work with a LABVIEW code to modify the control variables and view the resulting waveform. The circuit included a proportional term (K_c), an integral term (T_i), and a derivative term (T_d). These system parameters were adjusted to meet the desired performance specifications. Theoretical results were simulated using MATLAB and compared to the experimental data. The experimental results in the form of waveforms were gathered using LABVIEW software. The findings showed that the values of $K_c =$, $T_i = \times 10^{-4}$, and $T_d = \times 10^{-5}$ led to a percent overshoot of %, a settling time of $T_s =$ seconds, and a rise time of $T_r =$ seconds for the PID controller. These values fall within the desired performance specifications which means that the PID speed control system adequately regulated the rotational speed of the turntable. However, the settling time and settling time did not agree with the theoretical values. It was determined that besides the large amount of noise in our system the difference in the experimental and theoretical results could be attributed to the back EMF. By lowering the back EMF the corrected simulation agreed with the theoretical results. Overall, this experiment was performed to demonstrate an understanding of closed loop control systems.

II Introduction

In order to properly analyze and design a control system to satisfy performance specifications, the first step is to understand the underlying theory governing the system [1]. Specifically, this experiment relies on adjusting the parameters of a PID feedback loop analog circuit to control the rotational speed of a turntable to achieve a set of given performance specifications. Since a closed loop control system operates on the principle of comparing the error between the input and the output, there must exist a mathematical equation which relates the two parameters. More specifically, a PID feedback control system uses three terms - the proportional gain (K_c), integration gain ($\frac{K_c}{T_i}$), and derivative gain (K_cT_d) - to match a desired signal based on the measured difference between the input and output of the system. The input of the system in the case of this experiment is a repeated, alternating unit step function, $u(t)$. A single pulse of the unit step function can is defined as follows:

$$u(t) = \begin{cases} 0 & \text{if } t = 0 \\ \\ 1 & \text{if } t > 0 \end{cases} \tag{1}$$

While this function is given in terms of time, it must be transformed into the Laplace domain. The Laplace transform of the input unit step function can be performed as follows:

$$U(s) = \int_0^\infty u(t) \cdot e^{-st} dt = \int_0^\infty 1 \cdot e^{-st} dt = \frac{-1}{s}[e^{-st}]\Big|_0^\infty = \frac{-1}{s}[0-1] = \frac{1}{s} \tag{2}$$

The system is simply a response to the input which the controller tries to match. The ratio of the output to the input of the system, commonly referred to as the transfer function, can be obtained from the given block diagram (Figure 1). Since this particular system compares the output rotational speed to the input supplied voltage, the transfer function can be shown as:

$$T(s) = \frac{\omega(s)}{V(s)} \tag{3}$$

Where $\omega(s)$ represents the speed of the motor and $V(s)$ represents the input voltage. The derivation of this transfer function can be performed using Mason's Gain Formula as follows:

$$T(s) = \frac{\sum_{i=1}^{k} P_i \cdot \Delta_i}{\Delta} \tag{4}$$

Where P_i is the i^{th} forward path gain, Δ_i is the determinant of the i^{th} forward path, Δ is the determinant of the system, and k is the total number of forward paths. From the given block diagram, there is only one forward gain which is observed to be:

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

There is a feedback loop which has a gain of:

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

The determinant of the system is found by subtracting the sum of all loop gains from 1. The single loop is substituted into the formula as follows:

$$\Delta = 1 - L_1 = 1 + K_t G_c(s) G(s) \tag{7}$$

The determinant of the i^{th} forward path is the determinant for the portion of the block diagram which is not touching the i^{th} forward path. Since all portions of the block diagram are touching the forward path, the determinant of the forward path is simply shown as:

$$\Delta_1 = 1 - 0 = 1. \tag{8}$$

Therefore, the closed loop transfer function is:

$$T(s) = \frac{\omega(s)}{V(s)} = \frac{\sum P_k \cdot \Delta_k}{\Delta} = \frac{G_c(s)G(s)}{1 + K_t G_c(s) G(s)} \tag{9}$$

where

$$G(s) = \frac{K_m}{(L_a s + R_a)(Js + b) + K_b K_m} \approx \frac{K_m}{R_a(J_S + b) + K_b K_m}, \tag{10}$$

and

$$G_c(s) = K_c + \frac{(K_c/T_i)}{s} + (K_c T_d)s. \tag{11}$$

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

$$\text{Armature} = \frac{K_m}{L_a s + R_a}, \tag{12}$$

$$\text{Load} = \frac{1}{J_S + b}, \tag{13}$$

and

$$\text{Back EMF} = K_b. \tag{14}$$

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

$$\text{Torque constant}(K_m) = 16.2\,\text{OZ} - \text{IN/A}, \tag{15}$$

$$\text{DC armature resistance}(R_a) = 11.5\,\Omega, \tag{16}$$

$$\text{DC armature inductance}(L_a) = 0, \tag{17}$$

$$\text{Moment of inertia}(J) = 2.5\,\text{OZ} - \text{IN}^2, \tag{18}$$

$$b = 0, \tag{19}$$

$$\text{Back EMF constant}(K_b) = 12\,\text{V/KRPM}, \tag{20}$$

and

$$\text{Tachometer constant}(K_t) = 12\,\text{V/KRPM}. \tag{21}$$

The transfer function can be expanded by substituting values of $G(s)$ and $G_c(s)$:

$$T(s) = \frac{(K_c + \frac{(K_c/T_i)}{s} + (K_cT_d)s)(\frac{K_m}{R_a(Js+b)+K_bK_m})}{1 + K_t(K_c + \frac{(K_c/T_i)}{s} + (K_cT_d)s)(\frac{K_m}{R_a(Js+b)+K_bK_m})}. \tag{22}$$

This can be simplified by multiplying by the denominator of $G(s)$:

$$T(s) = \frac{(K_c + \frac{(K_c/T_i)}{s} + (K_cT_d)s)(\frac{K_m}{R_a(Js+b)+K_bK_m})}{1 + K_t(K_c + \frac{(K_c/T_i)}{s} + (K_cT_d)s)(\frac{K_m}{R_a(Js+b)+K_bK_m})} \cdot \frac{R_a(Js+b) + K_bK_m}{R_a(Js+b) + K_bK_m}. \tag{23}$$

The result of this operation is:

$$T(s) = \frac{K_m(K_c + \frac{(K_c/T_i)}{s} + (K_cT_d)s)}{R_a(Js+b) + K_bK_m + K_tK_m(K_c + \frac{(K_c/T_i)}{s} + (K_cT_d)s)}. \tag{24}$$

To simplify the complex fractions, the transfer function can be multiplied by $\frac{s}{s}$ as follows:

$$T(s) = \frac{K_m(K_c + \frac{(K_c/T_i)}{s} + (K_cT_d)s)}{R_a(Js+b) + K_bK_m + K_tK_m(K_c + \frac{(K_c/T_i)}{s} + (K_cT_d)s)} \cdot \frac{s}{s}. \tag{25}$$

The result of this operation is:

$$T(s) = \frac{K_m(K_cs + (K_c/T_i) + (K_cT_d)s^2)}{R_a(Js^2 + bs) + K_bK_ms + K_tK_m(K_cs + (K_c/T_i) + (K_cT_d)s^2)}. \tag{26}$$

Combining like terms yields:

$$T(s) = \frac{K_m(K_cs + (K_c/T_i) + (K_cT_d)s^2)}{s^2(R_aJ + K_tK_m(K_cT_d)) + s(R_ab + K_bK_m + K_tK_cK_m) + (K_tK_m(K_c/T_i))}. \tag{27}$$

This can be compared to the general form of a transfer function for a second order ordinary differential equation which can be represented as:

$$T(s) = \frac{k\omega_n^2}{s^2 + 2\zeta\omega_n^2 s + \omega_n^2} \tag{28}$$

Where ω_n is the undamped natural frequency and ζ is the damping ratio. However, the transfer function must be manipulated into the general form by isolating the s^2 term of the characteristic equation (the polynomial in the denominator). This isolation is accomplished by multiplying the entire transfer function by $\frac{1}{R_aJ+K_tK_mK_t}/\frac{1}{R_aJ+K_tK_mK_t}$ which is shown as follows:

$$T(s) = \frac{K_m(K_c s + (K_c/T_i) + (K_cT_d)s^2)}{s^2(R_aJ + K_tK_m(K_cT_d)) + s(R_ab + K_bK_m + K_tK_cK_m) + (K_tK_m(K_c/T_i))} \cdot \frac{\frac{1}{R_aJ+K_tK_mK_t}}{\frac{1}{R_aJ+K_tK_mK_t}}. \tag{29}$$

The result of this operation now gives the transfer function in the general form as presented in Eq. (28):

$$T(s) = \frac{K_m \frac{(K_c s+(K_c/T_i)+(K_cT_d)s^2)}{(R_aJ+K_tK_m(K_cT_d))}}{s^2 + s\frac{(R_ab+K_bK_m+K_tK_cK_m)}{(R_aJ+K_tK_m(K_cT_d))} + \frac{(K_tK_m(K_c/T_i))}{(R_aJ+K_tK_m(K_cT_d))}}. \tag{30}$$

The coefficients of the characteristic equation of the general form and this system's transfer function can be equated to determine ω_n and ζ. The characteristic equation of the general form and the transfer function of this system are set equivalent as follows:

$$s^2 + s\frac{(R_ab + K_bK_m + K_tK_cK_m)}{(R_aJ + K_tK_m(K_cT_d))} + \frac{(K_tK_m(K_c/T_i))}{(R_aJ + K_tK_m(K_cT_d))} = s^2 + 2\zeta\omega_n^2 s + \omega_n^2. \tag{31}$$

The value of ω_n can be determined as follows:

$$\omega_n = \sqrt{\frac{(K_tK_m(K_c/T_i))}{(R_aJ + K_tK_m(K_cT_d))}}. \tag{32}$$

Similarly, the value of ζ can be determined as follows:

$$\zeta = \frac{R_ab + K_bK_m + K_tK_cK_m}{2\omega_n(R_aJ + K_tK_m(K_cT_d))}. \tag{33}$$

To get the output of a transfer function, the function must be multiplied by the input. In this case, the Laplace transform of the unit step input was found to be $\frac{1}{s}$ in Eq.(2). For simplicity, the transfer function in the form of Eq.(27) is used. Therefore, the output speed of the motor is expressed as:

$$\omega(s) = T(s)V(s) = \frac{K_m(K_c s + (K_c/T_i) + (K_c T_d)s^2)}{s^2(R_a J + K_t K_m(K_c T_d)) + s(R_a b + K_b K_m + K_t K_c K_m) + (K_t K_m(K_c/T_i))} \cdot \frac{1}{s}. \tag{34}$$

The result of this operation is:

$$\omega(s) = \frac{K_m(K_c s + (K_c/T_i) + (K_c T_d)s^2)}{s^3(R_a J + K_t K_m(K_c T_d)) + s^2(R_a b + K_b K_m + K_t K_c K_m) + s(K_t K_m(K_c/T_i))}. \tag{35}$$

To isolate the numerator as a polynomial, the transfer function must be multiplied by $\frac{1/K_m}{1/K_m}$ as follows:

$$\omega(s) = \frac{K_m(K_c s + (K_c/T_i) + (K_c T_d)s^2)}{s^3(R_a J + K_t K_m(K_c T_d)) + s^2(R_a b + K_b K_m + K_t K_c K_m) + s(K_t K_m(K_c/T_i))} \cdot \frac{1/K_m}{1/K_m}. \tag{36}$$

The result of this operation is:

$$\omega(s) = \frac{K_c s + (K_c/T_i) + (K_c T_d)s^2}{s^3(\frac{R_A J}{K_m} + K_t(K_c T_d)) + s^2(\frac{R_A b}{K_m} + K_b + K_t K_c) + s(K_t(K_c/T_i))]}. \tag{37}$$

However, s can be factored out of the denominator which yields:

$$\omega(s) = \frac{K_c s + (K_c/T_i) + (K_c T_d)s^2}{s[(\frac{R_A J}{K_m} + K_t(K_c T_d))s^2 + (\frac{R_A b}{K_m} + K_b + K_t K_c)s + (K_t(K_c/T_i))}. \tag{38}$$

This expression of the rotational speed of the motor as a function of the Laplace variable can be transformed into the motor speed as a function of time using the inverse Laplace transform. To avoid solving the inverse Laplace transform of this transfer function with the gains as variables, the inverse Laplace will be determined once an experimental value of K_c, K_c/T_i, and $K_c T_d$ are determined. However, this is a necessary step since it enables the analysis of the time response of the system by transforming the system into a function of time. From that point, the experimental values of the parameters of interest can be determined to ascertain whether the system is considered to be within specifications or not.

To determine the gains of the PID controller, it is necessary to understand how the fundamental mathematics of the system are related to the physical interpretation of components specific to this experiment. For instance, the input signal of the system is provided by the waveform generator which 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 the DAQ so that the signal can be viewed from the LabView 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 83.3 RPM, which the PID controller tries to match by minimizing the error between the output and the input. 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 and is performed follows:

$$V_{out} - I_2 R_2 - 0 - V_{in} = 0 \tag{39}$$

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:

$$I_2 = \frac{V_{out} - V}{R_2} \tag{40}$$

The current across the resistor connected to the negative input of the op-amp is also necessary:

$$I_1 = \frac{V_{in} - 0}{R_1} \tag{41}$$

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

$$V_{out} = V_{in}(1 + \frac{R_2}{R_1}) \tag{42}$$

The potentiometer resistance, R_2, was solved to be 23.1kΩ which is connected from the negative terminal of the op-amp to its output, and a 1kΩ 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 differential amplifier where the difference between the output and the input is measured. This is important since 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 and adjust accordingly.

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 according to the given specifications. It is important to note that each of these performance metrics must be analyzed once the inverse Laplace transform of the transform function is determine since they are related to the time response of the output. Each of these metrics must be determined by analyzing the time response of the system after it is subjected to a single pulse of the unit step input. The rise time (T_r) is the time the system takes to initially reach the desired value. The settling time (T_s) is the time the system takes to fall within a specified percent of the final value. The percent overshoot (PO) is the maximum value the system reaches as a percentage of the desired value. In the case of this experiment, the design specifications were given to be a PO of less than 10% of the input, a settling time T_s to within 2% of the final value in less than 500 ms, and a rise time T_r less than 200 ms. PO, is calculated as follows:

$$PO = 100e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}}, \tag{43}$$

Manipulating to calculate the actual percentage overshoot of the signal, the expression is expressed as follows:

$$PO = 100(\frac{\omega_{max} - \omega_{settled}}{\omega_{settled}}), \tag{44}$$

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 calculated as follows:

$$T_s = \frac{ln(2/100)}{\omega_n\zeta}. \tag{45}$$

Rise time, T_r, is calculated as:

$$T_r = \frac{\pi - \cos^{-1}(\zeta)}{\omega_n\sqrt{1-\zeta^2}}. \tag{46}$$

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

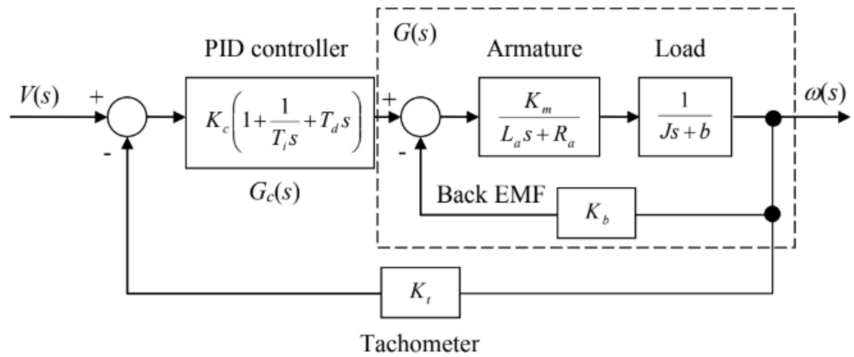

Figure 1 PID Speed Control System Block Diagram [2]

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 maintaining 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.

The LabView program was created as a means for controlling the parameters of the PID system as well as recording the input and output waveforms in addition to satisfying the requirement of extra design features. The LabView program relies on reading and writing voltage signals to the DAQ board. The Virtual Instrument (VI) DAQmx Create Virtual Channel was used to read analog input (AI) voltages by defining the DAQ pins 1 and 4 as input channels AI0 and AI1, respectively. These channels were defined in the reference single-ended configuration to measure the voltage signals with respect to ground. Similarly, the DAQmx Create Virtual Channel VI was used to write analog output (AO) voltages by defining pin 15 on the DAQ as output channel AO0. Physically, channel AI0 is connected to the waveform generator which is the setpoint of the system. Channel AI1 is the feedback from the tachometer which serves as the PID process variable. Channel AO0 is the output of the PID control algorithm within the LabView program. These voltage signals are read into the LabView program using the DAQmx Read VI in the analog one-dimensional waveform single sample for multiple channels configuration. The waveform data are displayed on waveform chart on the front panel of the program so the user can observe the data in real time. The data are also wired to the PID VI which uses channel AI0 as the setpoint and channel AI1 as the process variable. The parameters of the PID operation are controlled by the user in real time using the front panel. At this point, the result of the PID function is sent to an output file using the Write to Measurement File VI along with the voltage signals from channels A10 and AI1. The filename is controlled by the user using the front panel. There is also an "enable" button that controls whether or not the measurement file is being created. Additionally, there is an indicator on the front panel to display whether or not the measurement file is being written to. Concurrently, the output of the PID function is wired to the DAQmx Write VI in the analog double single channel single sample configuration which writes a voltage signal to previously defined output channel A0O. Simultaneously, the percent overshoot of the system is calculated by subtracting the setpoint from the process variable, dividing by the setpoint, taking the absolute value, multiplying that value by 100, displaying that value on the front panel, and displaying whether or not that value is wthin an acceptable range using an indicator on the front panel. All of these processes happen in a while loop which iterates so long as the "stop" button is not pressed. The cycle time is measured using the Tick Count VI which returns the loop time of the code in milliseconds and displays it on the front panel. This function relies on the use of shift register nodes at each end of the loop to actually measure how long the loop takes. There is also a safety feature in the code that detects the presence of the operator and halts the turntable if the operator is absent. This is accomplished using the Initialize Keyboard VI to define the keyboard as a device. Using said device, data is collected using the Acquire Input Data VI which determines what key is being pressed. If the spacebar key is not being pressed during operation, the program exits the loop and a voltage of zero is written to the motor.

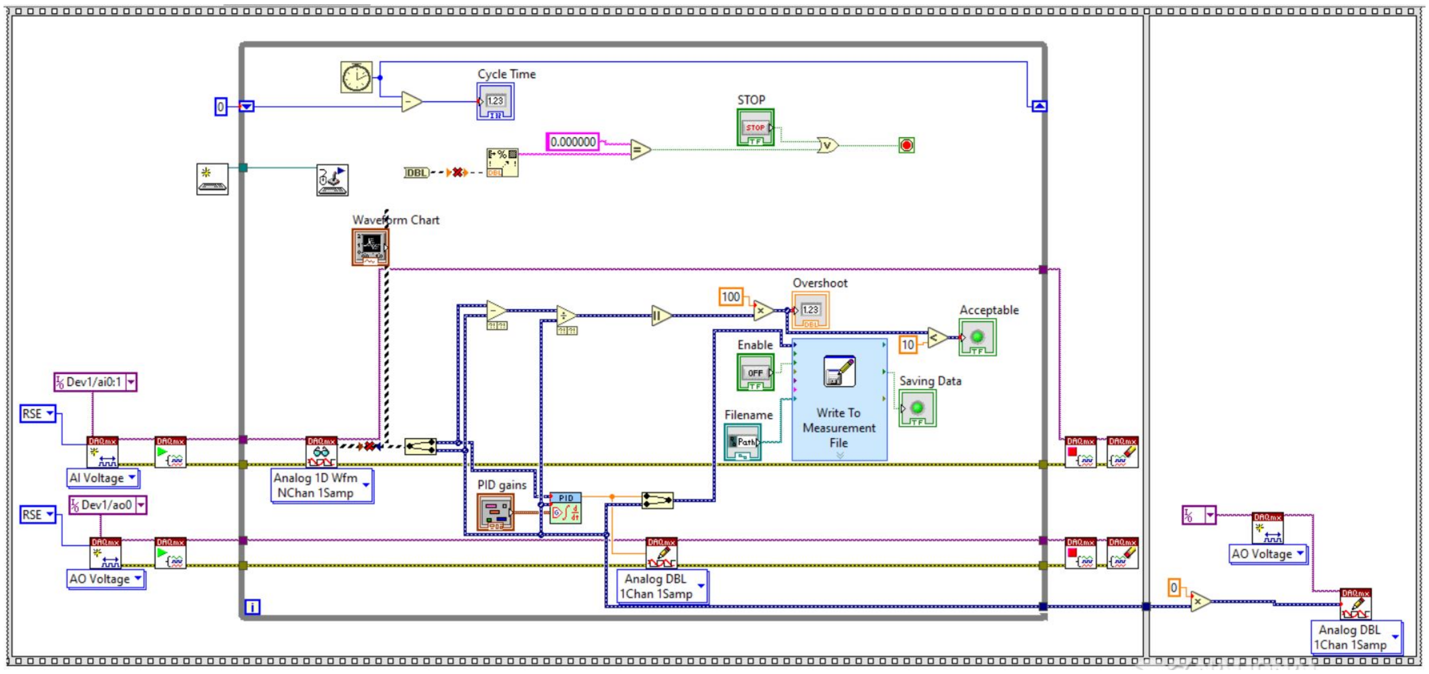

Figure 2 LabView Block Diagram Diagram [2]

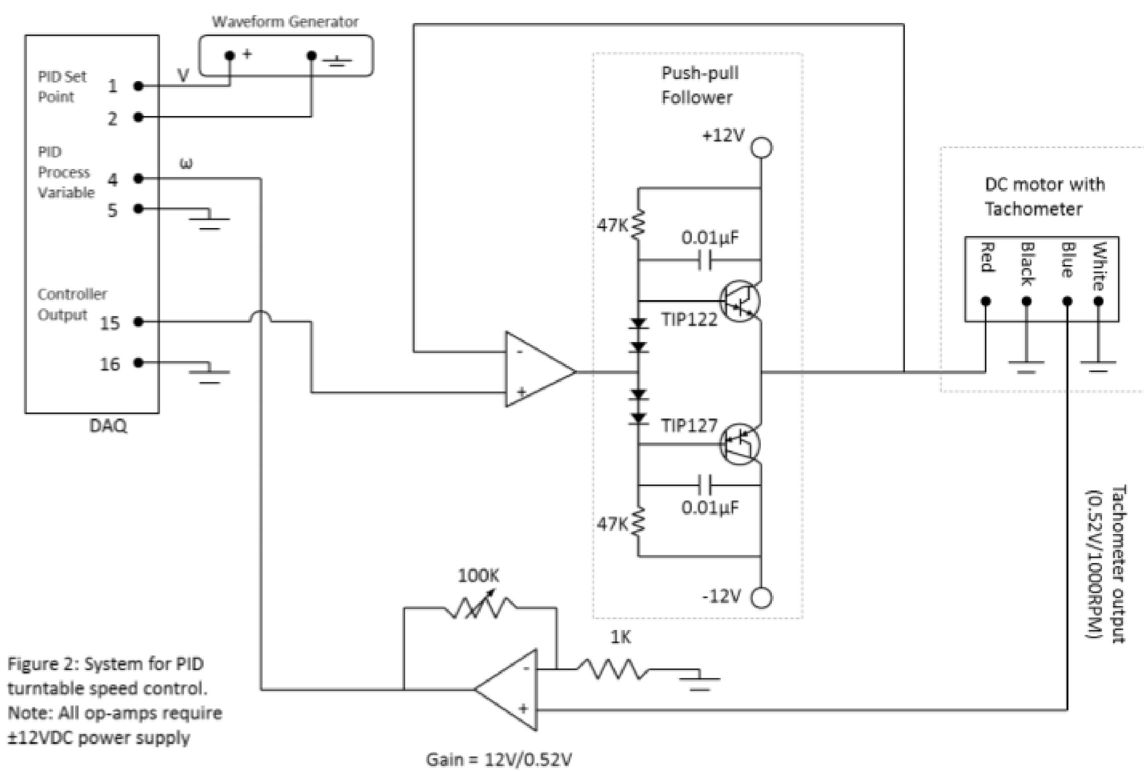

Figure 3 PID Speed Control System Circuit Diagram [2]

IV Results

As described in the Experimental Procedure (Section III), the digital circuit's PID terms were modified to perform as a proportional controller. While calibrating the P controller to match the square wave as closely as possible, the proportional gain parameter, K_c was increased until a point of marginal stability was observed at a value of 4. This experimental result is recorded in Experimental PID parameter Values (Table 1). Due to the fact that the signal experienced instability before reaching the desired square wave, there is a large gap between the desired and actual outputs as observed in experimental P controller input and output voltage versus time (Figure 4).

Similarly, the circuit's PID terms were then modified to make it a PI controller. While calibrating the PI controller, the PID parameters K_c and T_i were adjusted until a point of marginal stability was observed. The result was a T_i value of 0.0005 minutes and an K_c value of 4. The output signal of the PI controller was observed to be within an acceptable range for the desired square wave signal which is shown in Experimental PI Controller Input and Output Voltage Versus Time (Figure 5). With the exception of the rise time, the theoretical values are all within an acceptable uncertainty range compared to the experimental PI voltage values. This corresponding uncertainty range is calculated in Error Analysis (Section V). The PI controller can be observed graphically to achieve the specifications of having a settling time to within 2% of the final value less at 385.5 ms on the negative half, and a rise time is less than 200 ms (Figure 5). The success of the PI controller gives an approximated value for K_c and T_i for the PID controller.

Finally, the circuit's PID terms were modified to act as a PID controller. While calibrating the PID controller, the T_i and K_c parameters were kept as close to the values used in the PI controller as possible since the PI controller functioned within all specifications. Optimal behavior was observed while R_{p2} was equal to 4, T_i was 0.0005 minutes, and K_d was 0.00001 minutes (Table 1). The performance of the PID controller was an improvement upon that of the PI controller since there was an observed increase in stability as seen in Experimental PID Controller Input and Output Voltage Versus Time with Settling and Rise Time (Figure 6).

The experimental rise time, settling time, and percent overshoot were all determined graphically from the experimental data (Figure 6). The rise time (T_r) was observed to be 0.042 ± 0.0034 seconds from the start of the waveform. The settling time (T_s) was observed to be 0.3845 ± 0.0034 seconds from the start of the waveform. Lastly, the peak value of the controller was determined to be 1.0659 V. With a final settling point of 1 V, the percent overshoot (PO) of the output is calculated using Eq.(47), as shown below, and its uncertainty is calculated in Error Analysis (Section V):

$$PO = 100(\frac{1.0659 - 1}{1}) = 6.59 \pm 6.41\%. \tag{47}$$

Theoretical simulations of the P, PI, and PID controllers were generated using MATLAB to establish theoretical expectations for the experiment and create a model for the behavior of the system. The source code of the script used to produce the simulated outputs can be found the in the Appendix (Section IX)(Figures A1-A3). The code utilized the transfer function from Eq.(30) to solve for output speed of the motor ($\omega(s)$). All of the given constants are: 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 proportional gain parameter. All of the aforementioned constant values are documented in Parameters of the DC motor, the Tachometer, and the Load (Table 2). This function of the output speed of the motor is converted to a function of time using the inverse Laplace transform, which was accomplished by the *ilaplace* command in MATLAB. The results of the *ilaplace* command are as follows:

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

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

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

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

.(48)

It should be noted that both J and K_m are converted to SI units for this calculation.

The MATLAB code did not simulate the behavior of the experimental P controller output within its uncertainty for the majority of the wavelength, which was calculated in the error analysis (Section V). As observed in the MATLAB simulated and experimental P controller input and output voltage versus time (Figure 7), while the rise was predicted well, the simulated value was consistently greater than it experimental counter part. The MATLAB code simulated the behavior of the experimental PI controller output within the uncertainty with the exception of the experimental percent overshoot, as observed in the MATLAB simulated and experimental PI controller input and output voltage versus time graph (Figure 8). Lastly, the MATLAB code simulated the output of the PID controller correctly with the only exception being the simulation having, again, a greater percent overshoot, as observed in the MATLAB Simulated and Experimental PID Controller Input and Output Voltage Versus Time (Figure 9). Additionally, the simulated signal for all three cases is significantly more stable that that observed in the lab From this simulation theoretical values for rise time $T_{r-theory}$, percent overshoot PO_{theory}, and settling time $T_{s-theory}$ are graphically obtained for the PID controller. From the simulated data the following information was recorded: $T_{r-theory} = 0.0481 \pm 0.0683$ seconds, a peak of 1.0768 V, and $T_{s-theory} = 0.1102 \pm 0.0217$ seconds. Using Eq.(44) and a settled signal of 1 V, the percent overshoot can calculated to be $7.68\% \pm 2.2623$. The signal behavior of the PID controller (Table 3) shows that both the theoretical signal and the experimental signal are within the specification of our PID design parameters. The results of previous experiments with a digital circuit PID controller and an analog circuit PID controller should be compared to this experiments data. The previous experiment with a digitally controlled PID controller (Lab 1) resulted in a set of PID terms for each type of controller (Table 4). The analog circuit PID controller from lab 2 resulted in potentiometer (Table 5) values that correlate to the gain of the circuit just like the PID terms. These PID gains were found for the setups of each lab (Table 6). Lastly, the performance parameters for the PID controllers in each lab (Table 7) are presented to make a comparison between the three.

Table 1 Experimental Parameter Values for all Experimental Controllers

	K_c	$T_i[minutes]$	$K_d[minutes]$
P	4	0	0
PI	4	0.0005	0
PID	4	0.0005	.00001

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

Parameter	Value
K_m	16.2 OZ-IN / A
R_a	11.5 Ω
L_a	0
J	2.5 $OZ\text{-}IN^2$
b	0
K_b	12 V / KRPM
K_t	12 V / KRPM

Table 3 Performance Specifications of PID Controller

	Specification Limit	Experimental Value	Theoretical Value
$T_r[ms]$	200	42 ±3.4	48.1 ± 68.3
$T_s[ms]$	500	384.5 ±3.4	110.2 ± 21.7
P.O. [%]	10	6.59 ±6.41	7.68 ± 2.2623

Table 4 Lab 1 Experimental P, I, and D Terms for Each Configuration [5]

	K_c	T_i	T_d
P	2	0	0
PI	2	4.5×10^{-4}	0
PID	2	5.0×10^{-4}	5×10^{-5}

Table 5 Experimental Potentiometer Values for all Experimental Controllers [6]

	$R_{p2}[k\Omega]$	$R_i[k\Omega]$	$R_{d2}[k\Omega]$
P	4.683	0	0
PI	4.2	0.94	0
PID	4.5	0.91	.535

Table 6 PID Gain Values for all Experiments [5][6]

	Lab 1	Lab 2	Lab 3
K_p	2	4.5	4
$K_i[Hz]$	4×10^3	1.0989×10^5	133.3333
$K_d[s]$	10×10^{-5}	5.35×10^-7	0.0024

Table 7 PID Experimental Performance Parameters for all Experiments [5][6]

	Lab 1	Lab 2	Lab 3
$T_r[ms]$	79.9	12±1	42±3.4
$T_s[ms]$	89.0	21±1	384.5±3.4
PO [%]	2.13	4 ±0.0641	6.59 ±6.41

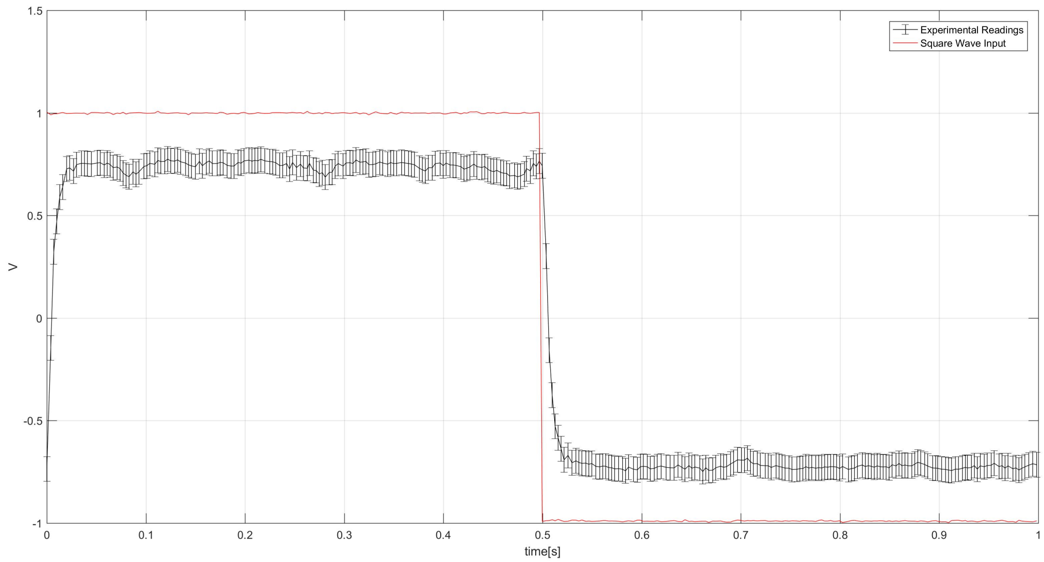

Figure 4 Experimental P Controller Input and Output Voltage Versus Time

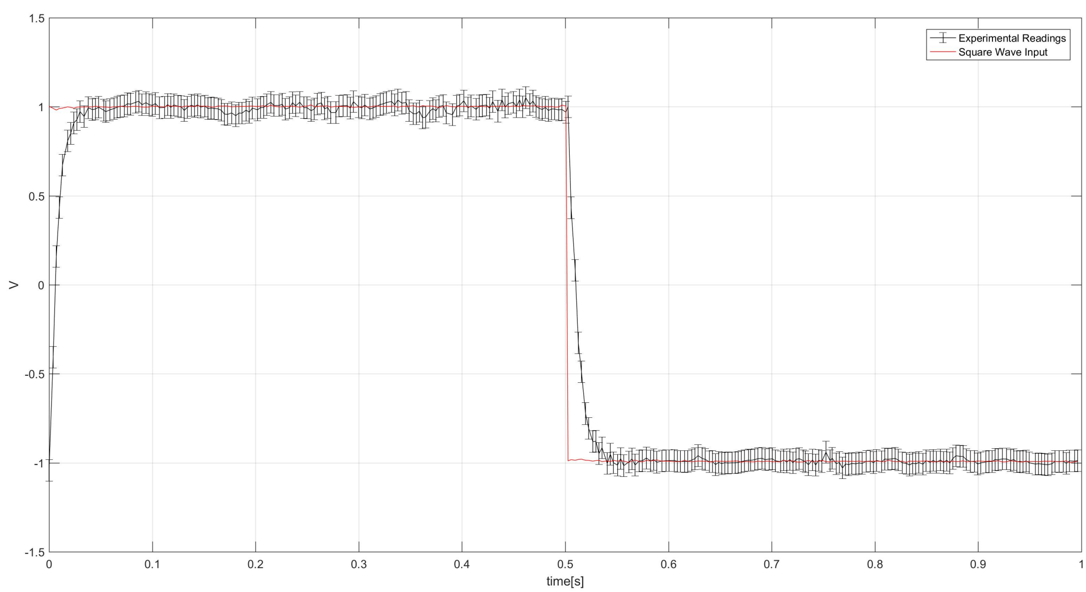

Figure 5 Experimental PI Controller Input and Output Voltage Versus Time

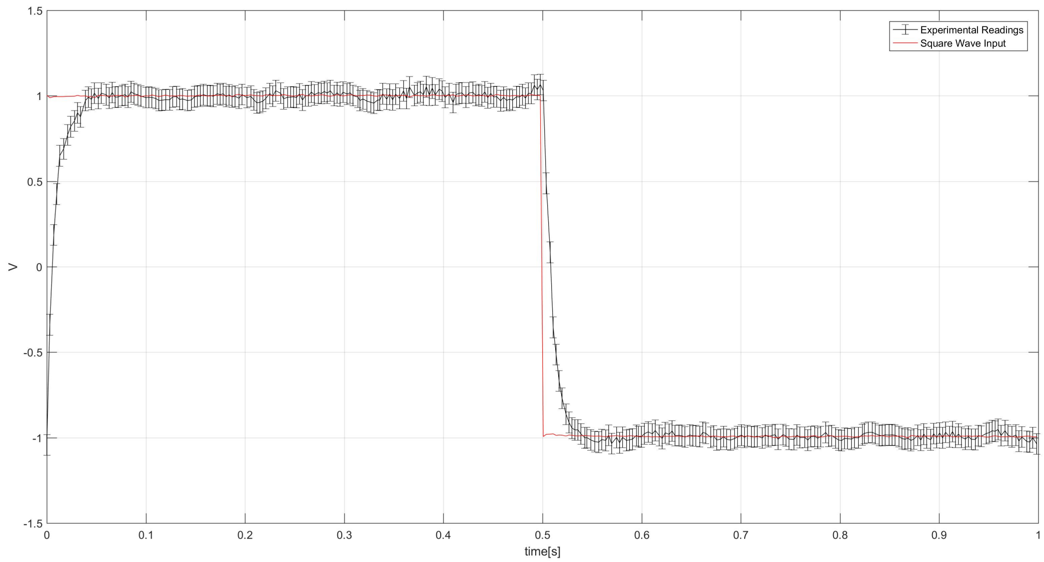

Figure 6 Experimental PID Controller Input and Output Voltage Versus Time with Settling Time and Rise Time

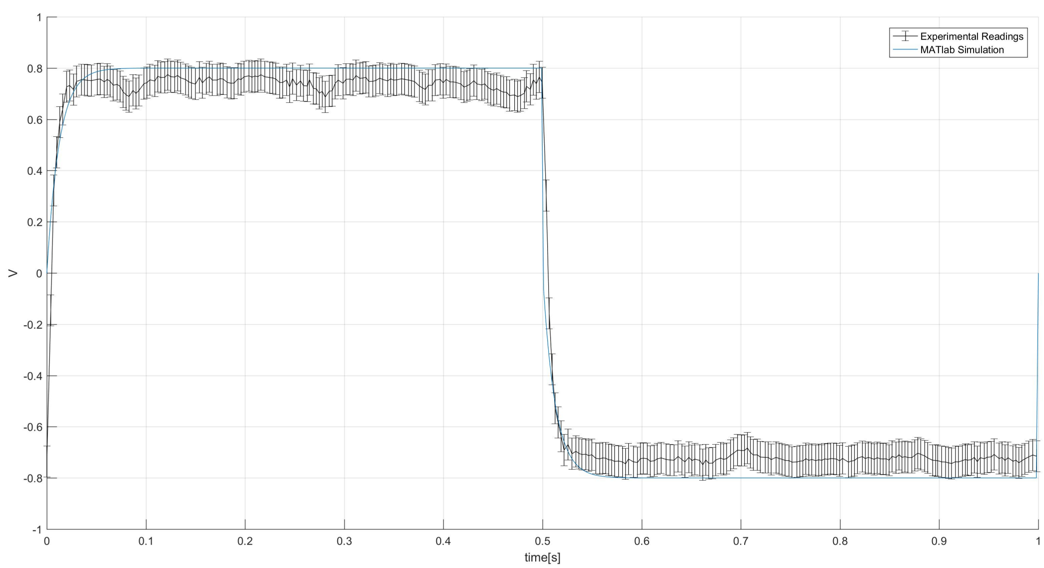

Figure 7 MATLAB Simulated and Experimental P Controller Input and Output Voltage Versus Time

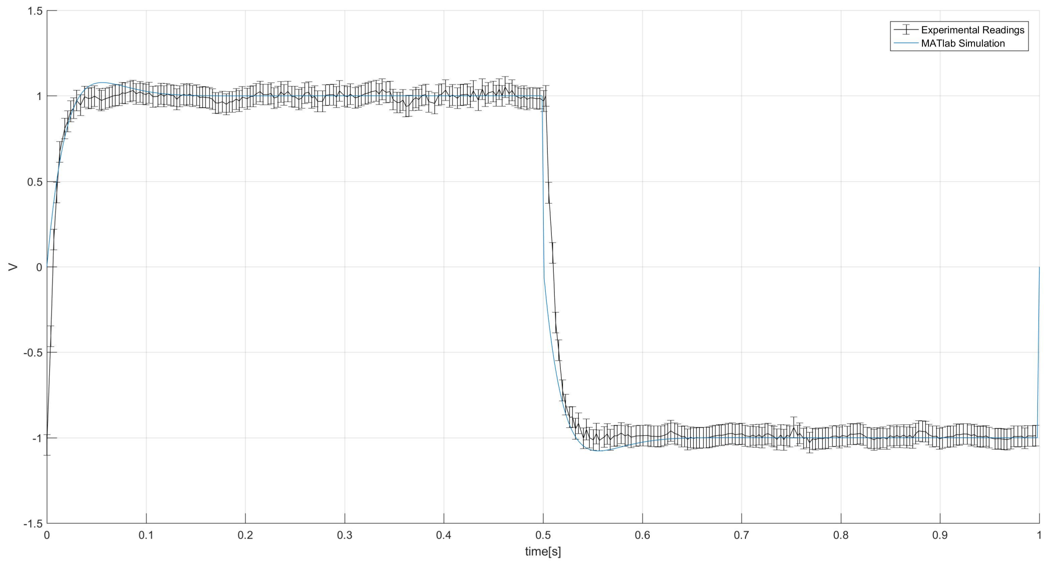

Figure 8 MATLAB Simulated and Experimental PI Controller Input and Output Voltage Versus Time

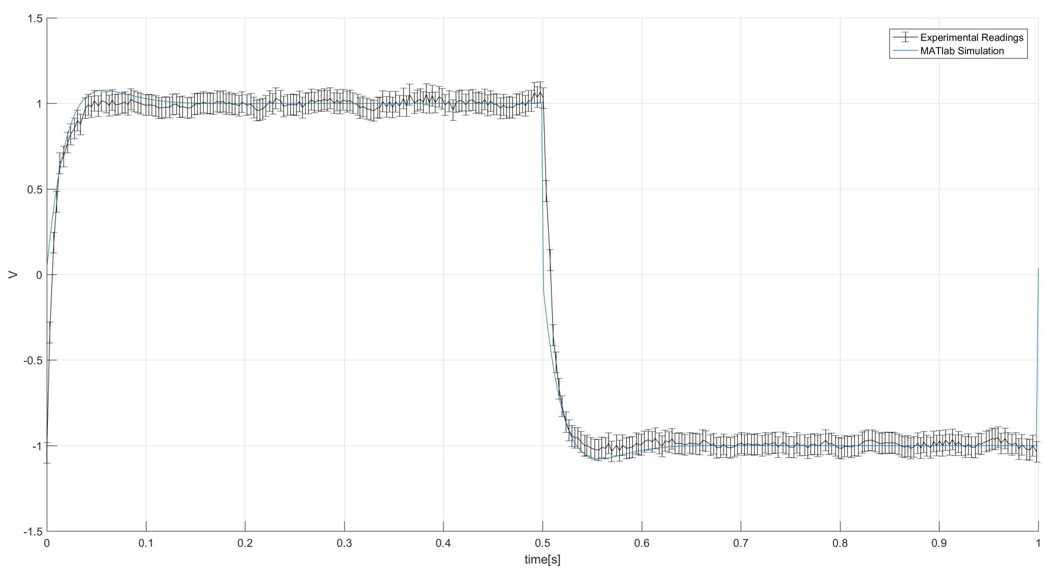

Figure 9 MATLAB Simulated and Experimental PID Controller Input and Output Voltage Versus Time

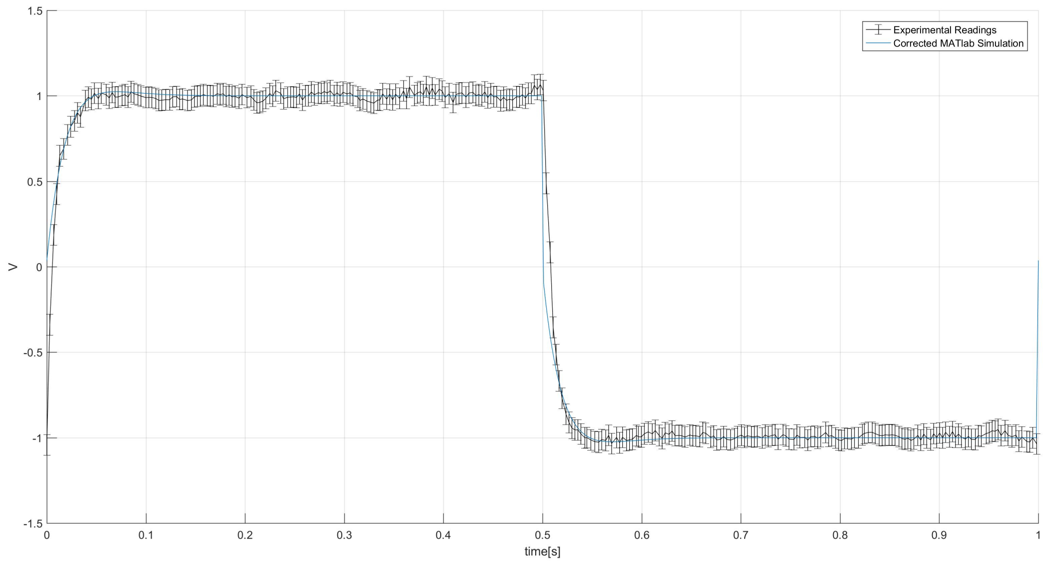

Figure 10 Corrected MATLAB Simulated and Experimental PID Controller Input and Output Voltage Versus Time

V Error Analysis

The experimental data gathered from this experiment was generated from a signal created with a waveform generator, which was used to operate an operational amplifier connected to the motor. A DC power supply was 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 0.0001 V should be applied to the waveform generator signal, ΔV_{wave-i} [7]. Additionally, the same datasheet points to a 2 mV accuracy of the generator as the manufacturer's error, so ΔV_{wave-m} of

0.002 V 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 0.01 V [8]. Next, 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 has a value of 0.06 V, 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 was connected through the DAQ system to the operational amplifier's positive input, which was replicated at the negative input connected to the motor. The power supply was connected to the motor when the transistor was 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. Lastly, it should be noted that the oscilloscope collected data at much lower resolution than that of the DAQ system for digital circuit of the previous experiment, taking measurements at $0.02V$ increments [5]. An additional ΔV_{osc} of $\pm 0.02V$ is considered. The input voltage error, ΔV, was calculated as follows:

$$\Delta V = \sqrt{\Delta V_{wave-i}^2 + \Delta V_{wave-m}^2 + \Delta V_{pow-i}^2 + \Delta V_{pow-m}^2 + \Delta V_{osc}}. \tag{49}$$

Substituting the known values, the ΔV term is solved to be $\pm 0.0641\,V$. This uncertainty is to be considered in the experimental results of all three controller configurations. ΔV is represented in the error bars of each graph containing experimental data (Figures 3-8).

In order to assess the uncertainty of the theoretical error values described by the MATLAB simulation, the mathematical elements of the transfer function, as expressed in Eq.(30), must be identified. The natural frequency (ω_n) and the damping ratio (ζ) are described by

Eq.(32) and Eq.(33), respectively. Both of those equations are derived from Eq.(28). These values are solved to be $\omega_n = 45.2886 \pm \Delta\omega_n$ Hz and $\zeta = 0.8492 \pm \Delta\zeta$. The uncertainty of ω_n and ζ can be propagated with the errors and corresponding partial derivatives of K_b, K_m, and R_a. The error ΔK_b is equal to $0.1K_b$, which is equal to $\pm 1.2\frac{V}{KRPM}$. Similarly, the error ΔK_m is equal to $0.1K_m$ which is equal to $\pm 1.62\frac{oz-in}{A}$ or $\pm 1.144X10^{-}2\frac{Nm}{A}$. Lastly, the error ΔR_a is equal to $0.15R_a$ which is equal to $\pm 1.725\Omega$. The partial derivative of ω_n with respect to K_m is shown as follows:

$$\frac{\partial \omega_n}{\partial K_m} = \frac{(K_c/T_i)K_tJR_a}{2\sqrt{\frac{(K_tK_m(K_c/T_i))}{(R_aJ+K_tK_m(K_cT_d))}}((K_cT_d)K_mK_t + JR_a)^2} \tag{50}$$

The partial derivative of ω_n with respect to R_a is shown as follows:

$$\frac{\partial \omega_n}{\partial R_a} = -\frac{(K_c/T_i)K_tJK_m}{2\sqrt{\frac{(K_tK_m(K_c/T_i))}{(R_aJ+K_tK_m(K_cT_d))}}((K_cT_d)K_mK_t + JR_a)^2}. \tag{51}$$

By obtaining the root sum square of the product of each partial derivative and each of the corresponding uncertainties, the total error in ω_n can be determined as follows:

$$\Delta\omega_n = \sqrt{(\Delta K_m \cdot \frac{\partial \omega_n}{\partial K_m})^2 + (\Delta K_m \cdot \frac{\partial \omega_n}{\partial R_a})^2} \tag{52}$$

Numerically, $\Delta\omega_n$ is solved to be $\pm 3.9315\,Hz$. Similarly, the partial derivatives of ζ in terms of ω_n, K_b, R_a, and K_m must be determined. The partial derivative of ζ with respect to ω_n is shown as follows:

$$\frac{\partial \zeta}{\partial \omega_n} = -\frac{K_bK_m + K_tK_cK_m}{2\omega_n^2(R_aJ + K_tK_m(K_cT_d))}, \tag{53}$$

The partial derivative of ζ with respect to K_b is shown as follows:

$$\frac{\partial \zeta}{\partial K_b} = \frac{K_m}{2\omega_n(R_aJ + K_tK_m(K_cT_d))}, \tag{54}$$

The partial derivative of ζ with respect to R_a is shown as follows:

$$\frac{\partial \zeta}{\partial R_a} = \frac{J(K_b K_m + K_t K_c K_m)}{2\omega_n (R_a J + K_t K_m (K_c T_d))^2}, \tag{55}$$

The partial derivative of ζ with respect to K_m is shown as follows:

$$\frac{\partial \zeta}{\partial K_m} = \frac{R_a J(K_b + K_t K_c)}{2\omega_n (R_a J + K_t K_m (K_c T_d))^2}. \tag{56}$$

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

$$\Delta\zeta = \sqrt{(\Delta\omega_n \cdot \frac{\partial \zeta}{\partial \omega_n})^2 + (\Delta K_b \cdot \frac{\partial \zeta}{\partial K_b})^2 + (\Delta R_a \cdot \frac{\partial \zeta}{\partial R_a})^2 + (\Delta K_m \cdot \frac{\partial \zeta}{\partial K_m})^2} \tag{57}$$

Numerically, $\Delta\zeta$ is calculated as ± 0.1657. Rise time, T_r, is defined as:

$$T_r = \frac{\pi - \cos^{-1}(\zeta)}{\omega_n \sqrt{1-\zeta^2}}. \tag{58}$$

The partial derivative of T_r with respect to ω_n and ζ must be determined. The partial derivative of T_r with respect to ω_n is shown as follows:

$$\frac{\partial T_r}{\partial \omega_n} = -\frac{\pi - \cos^{-1}(\zeta)}{\omega_n^2 \sqrt{1-\zeta^2}} \tag{59}$$

$$\frac{\partial T_r}{\partial \zeta} = \frac{1}{\omega_n (1-\zeta^2)} + \frac{\zeta\,(\pi - \cos^{-1}(\zeta))}{\omega_n (1-\zeta^2)^{3/2}}, \tag{60}$$

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:

$$\Delta T_r = \sqrt{(-\frac{\pi - \cos^{-1}(\zeta)}{\omega_n^2 \sqrt{1-\zeta^2}}\Delta\omega_n)^2 + ((\frac{1}{\omega_n (1-\zeta^2)} + \frac{\zeta\,(\pi - \cos^{-1}(\zeta))}{\omega_n (1-\zeta^2)^{3/2}})\Delta\zeta)^2}. \tag{61}$$

Numerically, the error in the rise time equates to ± 0.0683 seconds.

The percent overshoot formula, PO, is defined as follows [1]:

$$PO = 100e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}}. \tag{62}$$

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

$$\frac{\partial PO}{\partial \zeta} = \frac{100\pi e^{-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}}}{(1-\zeta^2)^{3/2}}. \tag{63}$$

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

$$\Delta PO = \frac{100\pi e^{-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}}}{(1-\zeta^2)^{3/2}}\Delta\zeta \tag{64}$$

The error, $\Delta P.O.$, solves to be $\pm 2.2623\%$.

Next, the settling time is the time when the value settles to 2% of the expected value and it is represented as follows [1]:

$$T_s = \frac{ln(2/100)}{\omega_n \zeta}. \tag{65}$$

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

$$\frac{\partial T_s}{\partial \omega_n} = -\frac{ln(2/100)}{\omega_n^2 \zeta} \tag{66}$$

and

$$\frac{\partial T_s}{\partial \zeta} = -\frac{ln(2/100)}{\omega_n \zeta^2} \tag{67}$$

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

$$\Delta T_s = \sqrt{(-\frac{\Delta\omega_n ln(2/100)}{\omega_n^2 \zeta})^2 + (-\frac{\Delta\zeta ln(2/100)}{\omega_n \zeta^2})^2}. \tag{68}$$

Numerically, the error in the settling time solves to be ± 0.0217 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.

By observing the time values for each recorded data point in the PID controller, it was observed to have an average time increment of 3.4 milliseconds. As a result, the experimental rise and settling time are attributed an uncertainty of 3.4 ms of incremental error. Additionally, the experimental percent overshoot is calculated to the peak recorded voltage, which is subjected to the aforementioned experimental voltage error present in (Figures 2-8) The experimental percent overshoot error can thereby be calculated using equation 51 replacing the $\omega_{settled} - \omega_{max}$ term with the voltage error $\pm 0.0641\,V$. The resulting uncertainty for experimental percent overshoot is 6.41 %.

VI Discussion

The MATLAB simulation for the PID controller implemented in this experimented replicated the percent overshoot of the experimental results, determined as $7.68\% \pm 2.26\%$, within the uncertainty which was determined as $6.59 \pm 6.41\%$. Additionally, the rise time was accurately simulated with a theoretical value of 48.1 ± 68.3 milliseconds within an experimental uncertainty of 42 ± 3.4 milliseconds. However, the settling time of the experimental data is found outside of the theoretical uncertainty. The theoretical value for settling time is 110.2 ± 21.7 milliseconds while the experimental value is 384.5 ± 3.4 milliseconds. Not only was the settling time incorrectly predicted for this control model, but it is significantly higher than the settling times achieved in lab 1 and 2. This was the result of a surplus of unexpected noise in the data observed during the lab. This noise was not eliminated by simplifying the circuit or by ensuring more solid electrical connections. The control system was designed to be the same as that of lab 1, however this noise prevented the same results for being achieved. The results recorded for the PID controller (Figure 6) are the best that could be achieved while trying to dampen the violent noise using the control parameters. This is why different control parameters had to be used from lab 1. It should be noted however, the voltage error associated with the error bars of the data would indicate a $\pm 2\%$ value from the square wave at any given point, thus any speculation on the settling time is largely speculative anyway as the data would indicate an immediate settling time as well.

In order to develop the simulation to more accurately describe the system in lab 1, we explored the possibility of the systems perhaps having an abnormally high back EMF. The same assumptions are made in the corrected simulation of the PID data for this lab (Figure

10) by utilizing a back EMF of 19 V/KRPM. As observed in the corrected simulation, a much better fit is made with the experimental data, the same effect the correction had in lab 1. As a result, both lab 1 and 3 indicate some abnormality in back EMF.

To compare the PID parameters from digital and analog control circuits, both are considered in terms K_p, (K_i), and (K_d) as shown in Lab 1 Experimental Proportional, Integral, and Derivative Time Terms for all Experimental Controllers (Table 4). For lab 2's analog circuit the parameters were calculated using the resistance and capacitance values found on (Table 5). As a result, K_p was found to be 4.5, (K_i) was found to be 1.0989×10^5 Hz, and (K_d) was found to be 5.35×10^{-7} seconds for the analog control circuit. To convert the digital control PID controller parameters to K_p, (K_i), and (K_d) the following equations were used:

$$K_p = K_c, \tag{69}$$

$$K_i = \frac{K_c}{T_i}, \tag{70}$$

and

$$K_d = K_c T_d. \tag{71}$$

By plugging in the values of time terms received from the PID controller with the digital controls for lab 1 (Table 4) the results for lab 1 parameters were as follows; K_p was found to be 2, (K_i) was found to be 4×10^3 Hz, and (K_d) was found to be 10×10^{-5} seconds for the digital control circuit. Once again, by plugging in the values time terms received from the PID controller with the digital controls for lab 3 (Table 1) the results for lab 3 parameters were as follows; K_p was found to be 4, (K_i) was found to be 133.33 Hz, and (K_d) was found to be 0.0024 seconds for the digital control circuit. These values are the PID gains for all the experiments (Table 6). It can be observed that the lab 3 control system as a much greater influence from the derivative control that the other two, with less influence from the integral control. Observing the performance parameters of all three experiments (Table 7) this lab, lab 3, contained the worse results. the source of the low quality results, as previously explained, was the large volume of noise present in the lab. If this noise was not a factor, we would expect similar results to lab 1, as the procedure only differs with the efficiency of the LABVIEW code. Overall, lab 1 produce the best PID controller.

In terms of cost, the system's physical component have a negligible cost since they contain only basic electrical components. The analog controlled PID controller in lab two needed more electrical components, such as potentiometers and therefore would cost more, although that increase is also basically negligible. However, lab two needed an oscilloscope which is significantly more expensive than a LABVIEW license, which runs 100 dollars annually. Over time the licensing costs would meet the cost of the oscilloscope. For training, this system is comprised of simple knobs to adjust the control parameters. Lab two would require the most training to use by adjusting the potentiometer and measuring the value for resistance instead of simply just changing a number in a LABVIEW code like was done in labs one and three. The maintenance of all three labs are don't seem to be very different.

VII Conclusions

- The experimental results showed that a PID turntable speed control system with digital controls can be created with accurate results.
- The control system used for Lab 1 and 2 was found to be better.

- The relevant results were found to be the Experimental Potentiometer Values for all Experimental Controllers (Table 1) and the Time Response of the PID Controller (Table 3).

- The experimental results did not agree with the theoretical values within acceptable ranges of uncertainty.

- 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. Additionally, there was a large amount of noise for which the source could not be determined.

- In future investigations, it is recommended to learn more about LABVIEW coding in general. Not how to put the code together but more information on how these tools actually work to analyze the results better.

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 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] Jeremy Nielsen, Tara Boyle, Scott Miller,"Lab 1: Digital PID Speed Control of a Turntable"

[6] Jeremy Nielsen, Tara Boyle, Scott Miller,"Lab 2: PID Speed Control of a Turntable"

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

[8] "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=4%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)
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('Lab3P','D147:D438');%400 693 694 992
Swave=xlsread('Lab3P','F147:F438');
time=xlsread('Lab3P','C147:C438');
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+(sqrt((0.005*12)^2+.01^2+0.002^2+0.0001^2))


 errorbar(time,PID,dPID,'k');
 hold on
 plot(t,V*12/1000);
 legend('Experimental Readings','MATlab Simulation')
  xlabel('time[s]')
  ylabel('V')
 figure(2)
 grid on
```

```
 errorbar(time,PID,dPID,'k');
hold on
plot(time,Swave,'r')
 hold on
xlabel('time[s]')
ylabel('V')

 grid on
legend('Experimental Readings','Square Wave Input')
```

Figure A1 MATLAB code for Experimental and Theoretical P Controller Plots

```
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=4%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=0.0000*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)
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('Lab3PI','D155:D447');%400 693 694 992
Swave=xlsread('Lab3PI','F155:F447');
time=xlsread('Lab3PI','C155:C447');
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+(sqrt((0.005*12)^2+.01^2+0.002^2+0.0001^2))


 errorbar(time,PID,dPID,'k');
 hold on
 plot(t,V*12/1000);
 legend('Experimental Readings','MATlab Simulation')
  xlabel('time[s]')
 ylabel('V')
 figure(2)
 grid on
  errorbar(time,PID,dPID,'k');
 hold on
```

```
plot(time,Swave,'r')
 hold on
xlabel('time[s]')
ylabel('V')


 grid on
legend('Experimental Readings','Square Wave Input')
```

Figure A2 MATLAB code for Experimental and Theoretical PI Controller Plots

```
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=4%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=0.00001*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('Lab3PID','D694:D992');%400 693 694 992
Swave=xlsread('Lab3PID','F694:F992');
time=xlsread('Lab3PID','C694:C992');
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+(sqrt((0.005*12)^2+.01^2+0.002^2+0.0001^2))


 errorbar(time,PID,dPID,'k');
 hold on
 plot(t,V*12/1000);
 legend('Experimental Readings','MATlab Simulation')
  xlabel('time[s]')
 ylabel('V')
 figure(2)
 grid on
  errorbar(time,PID,dPID,'k');
 hold on
```

```
 plot(time,Swave,'r')
  hold on
 xlabel('time[s]')
 ylabel('V')
 grid on
legend('Experimental Readings','Square Wave Input')




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=4%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=0.00001*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)


PID=xlsread('Lab3PID','D694:D992');%400 693 694 992
Swave=xlsread('Lab3PID','F694:F992');
time=xlsread('Lab3PID','C694:C992');
time=time-time(1)

difference=abs((Swave-PID));
S = stepinfo(PID,time)

 errorbar(time,PID,dPID,'k');
 hold on
 plot(t,(V)*12/1000);
 legend('Experimental Readings','Corrected MATlab Simulation')
  xlabel('time[s]')
 ylabel('V')
```

Figure A3 MATLAB code for Experimental and Theoretical PID Controller Plots

Comment Summary
Page 3

1. numbers missing

Page 18

2. Caption should be attached to the figure.