# Laboratory Experiment II : Implementation of a Speed Controller

MTRN3020 - Modelling and Control of Mechatronic Systems

I verify that the contents of this report are my own work

Zachary Hamid
z5059915
06/10/2017

## Contents

## 1 Introduction

The purpose of this experiment was to design and implement a discrete controller and use it to control the speed of a motor that was connected to a generator with both no load resistors and a sequence of different load resistors attached in parallel to each other. To determine the validity of the controller, experimental data was used as a comparison to simulated data generated by a Simulink diagram that utilised the designed controller.

The first part of the experiment (Part A) was primarily for design verification to check the validity of the designed controller. The second part of the experiment (Part B) was to ensure that the controller was able to account for disturbances in the load applied to the motor generator system that were caused by varying the number of resistors connected.

## 2 Aim

The aim of this laboratory experiment was to maintain a desired motor speed, $\omega_d$, using a speed controller implemented into a motor generator system that was designed using Ragazzini's method of direct analytical design. This speed had to be maintained regardless of the number of resistors that were connected and changes in number of resistors connected, which determined the load torque that the motor generator system experienced. The schematic of motor generator system is below:

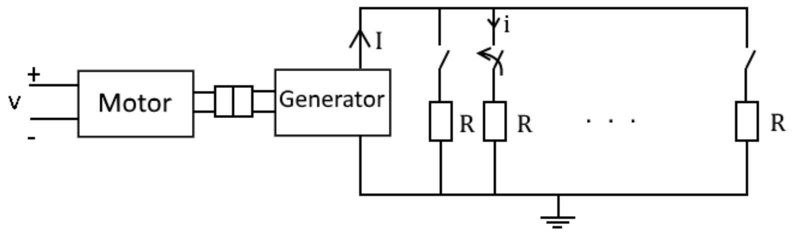

Figure 1: Schematic of motor generator system used in experiment

## 3 Experimental Procedure

There were two parts to the experimental procedure.
For the first part of the experiment, the motor was initially run at 1000 rpm, and at a specific time, the motor speed was suddenly changed to 2000 rpm, representing a step input.
For the second part of the experiment, there were 8 sets of 200 consecutive data samples (totalling 1600 data samples), where each set had a different number of resistors connected to the generator. The first and last sets had no resistors connected and the 6 sets in the middle had a number of resistors connected that were determined by the hexadecimal representation of my student number, given in MATLAB by dec2hex(5059915)= 4D354B, which gave a unique resistor sequence of:

$$0\rightarrow4\rightarrow13\rightarrow3\rightarrow5\rightarrow4\rightarrow11\rightarrow0$$

## 4 Controller Design

To find the required discrete transfer function of the controller mentioned previously, the first thing that must be done is obtain some constants from experimental data related to the system. This is done using MATLAB's curve-fitting tool, *cftool*. By fitting to the data obtained from *noload.m* [motor speed (counts/s) against time (s)] using a fit equation of $f(t) = v(t) * A(1 - e^{-Bt})$, where $B = \frac{1}{\tau}$ and $v(t) = 24$V, the following fit was found:

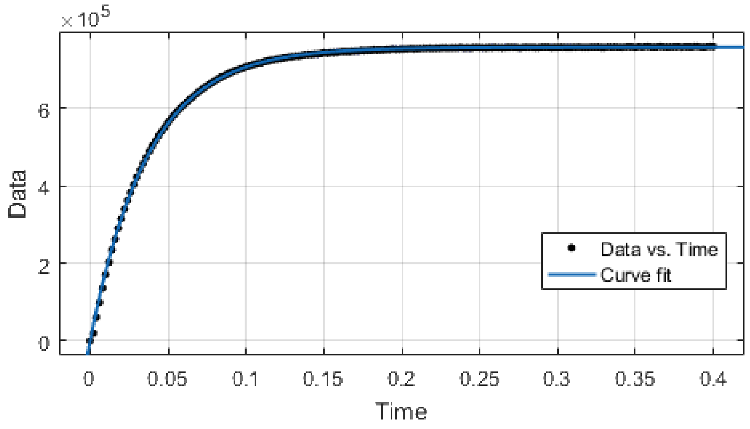

Figure 2: MATLAB *cftool* curve-fit to *noload.m* data

It can be seen that the *cftool* gives a good fit to the data, and so the constants obtained from this will be accurate enough for the controller design. From MATLAB's *cftool*, the constants were found to be:

$$A = 31590 \text{ counts/s/V} \tag{1}$$

$$B = 26.99 \rightarrow \tau = \tfrac{1}{26.99} = 0.0370507595 \text{ s} \tag{2}$$

Using the above results, a transfer function relating the applied voltage and the speed (in counts/s) can be derived as:

$$G_{P1}(s) = \frac{A}{1 + \tau s} \tag{3}$$

However, we can only sample the counts not the counts/s and so an integrator must be added to make the transfer function relate the applied voltage to encoder counts and so the final plant is:

$$G_P(s) = \frac{A}{s(1 + \tau s)} \tag{4}$$

From the block diagram given in the lab document[1], combining all blocks to the right of the controller block, the following expression is obtained:

$$G_P(z) = \mathcal{Z}\left[\frac{24}{126}\frac{(1 - e^{-sT})}{s}G_P(s)\frac{(z-1)}{zT}\right] \tag{5}$$

Where $T = 0.01$s (from individual design data), simplifying this expression gives:

$$G_P(z) = \frac{24}{126}\frac{(z-1)}{zT}(1-z^{-1})\mathcal{Z}\left[\frac{G_P(s)}{s}\right] = \frac{24A}{126}\frac{(z-1)^2}{z^2T}\mathcal{Z}\left[\frac{1}{s^2(1+\tau s)}\right] \tag{6}$$

Simplifying inside the z-transform brackets so the z-transform tables can be used:

$$G_P(z) = \frac{24A}{126}\frac{(z-1)^2}{z^2T}\mathcal{Z}\left[\frac{1/\tau}{s^2(s+1/\tau)}\right] \tag{7}$$

Applying the z-transform tables for the form $\dfrac{a}{s^2(s+a)}$:

$$G_P(z) = \frac{24A}{126}\frac{(z-1)^2}{z^2T}\frac{z[((T/\tau)-1+e^{-T/\tau})z + (1-e^{-T/\tau}-(T/\tau)e^{-T/\tau})]}{(1/\tau)(z-1)^2(z-e^{-T/\tau})} \tag{8}$$

Simplifying the above, setting $\alpha = T/\tau$, $\beta = e^{-T/\tau}$, and factorising the numerator:

$$G_P(z) = \frac{24A}{126\alpha}(\alpha-1+\beta)\frac{(z+\frac{(1-\beta-\alpha\beta)}{(\alpha-1+\beta)})}{z(z-\beta)} \tag{9}$$

Calculating the zero above using result (2) and $T = 0.01$ s (from individual design data) to determine if it is ringing or not:

$$\frac{(1-\beta-\alpha\beta)}{(\alpha-1+\beta)} = \frac{0.0304874337933027}{0.03335583617322} = 0.9140059818911 \tag{10}$$

This is a positive value and $< 1$, therefore it is a stable, ringing zero and must be eliminated in the numerator $F(z)$ so it isn't cancelled in the plant. Calculating the pole of (9) to determine if it is ringing:

$$-\beta = -e^{-T/\tau} = -0.76345583587768 \tag{11}$$

From the above result, it is $> -1$ and $< 0$ and so it can be seen that the pole of (9) is stable and not ringing, and so does not need to be included in $F(z)$. Keeping these variables to ensure as much accuracy as possible by eliminating carry-through rounding errors, to determine $F(z)$, we first find the desired closed-loop s-domain root $s = -\frac{1}{\tau_d}$, where $\tau_d = 0.046$s (from individual design data) and so the z-domain root is:

$$z = e^{sT} = e^{-T/\tau_d} \tag{12}$$

From this result we have the starting point for $F(z)$ as:

$$F(z) = \frac{b_0(z+z_1)}{(z-e^{-T/\tau_d})} \tag{13}$$

Where $z_1$ is the ringing zero from (10). Now, checking whether the causality constraint holds; the pole-zero deficiency of $F(z)$, $\mathcal{D}\{F(z)\} = 0$, but the pole-zero deficiency of $G_P(z)$, $\mathcal{D}\{G_P(z)\} = 1$. Since $\mathcal{D}\{F(z)\} < \mathcal{D}\{G_P(z)\}$, the causality constraint is violated. Therefore the new $F(z)$ must be:

$$F(z) = \frac{b_0(z+z_1)}{z(z-e^{-T/\tau_d})} \tag{14}$$

Determine the last unknown, $b_0$, using the desire of zero steady-state error, that is $F(z) = 1$, this happens at $z = 1$ and therefore:

$$F(1) = \frac{b_0(1 + z_1)}{(1 - e^{-T/\tau_d})} = 1 \rightarrow b_0 = \frac{(1 - e^{-T/\tau_d})}{(1 + z_1)} \tag{15}$$

The above will be left as $b_0$ for now and calculated later to retain accuracy of the result. Now everything required to design the controller has been found, we can now derive the controller equation using the following expression (by Ragazzini's method):

$$G_c(z) = \frac{1}{G_p(z)} \frac{F(z)}{1 - F(z)} \tag{16}$$

Substituting all known values, the following is obtained:

$$G_c(z) = \frac{126\alpha}{24A(\alpha - 1 + \beta)} \frac{z(z - \beta)}{(z + \frac{(1-\beta-\alpha\beta)}{(\alpha-1+\beta)})} \frac{\frac{b_0(z+z_1)}{z(z-e^{-T/\tau_d})}}{(1 - \frac{b_0(z+z_1)}{z(z-e^{-T/\tau_d})})} \tag{17}$$

For ease of simplification (and to retain accuracy), setting $C = \frac{126\alpha}{24A(\alpha-1+\beta)}$, $D = z_1 = \frac{(1-\beta-\alpha\beta)}{(\alpha-1+\beta)}$, $E = e^{-T/\tau_d}$, then (17) becomes:

$$G_c(z) = C\frac{z(z - \beta)}{(z + D)} \frac{\frac{b_0(z+D)}{z(z-E)}}{(1 - \frac{b_0(z+D)}{z(z-E)})} \tag{18}$$

This simplifies down to:

$$G_c(z) = Cb_0 \frac{z^2 - \beta z}{z^2 - (E + b_0)z - b_0 D} \tag{19}$$

And so, substituting all known values, the final discrete transfer function for the controller is:

$$\boxed{G_c(z) = \frac{M(z)}{E(z)} = \frac{0.000137274076028649z^2 - 0.000104802694458788z}{z^2 - 0.906696735981136z - 0.0933032640188645}} \tag{20}$$

This is the controller block used in the Simulink diagram (shown in the next section). To get the controller equation used in the experimental data, we divide the above through by $z^2$, rearrange for M(z) and take the inverse z-transform to get:

$$\begin{aligned} m(k) = 0.906696735981136 * m(k-1) + 0.0933032640188645 * m(k-2) +& \\ 0.000137274076028649 * e(k) - 0.000104802694458788 * e(k-1)& \end{aligned} \tag{21}$$

Where $m(k)$ is the current PWM value, $m(k-1)$ is the previously sampled PWM value, $m(k-2)$ is the PWM value before that, $e(k)$ is the currently measured error in speed, and $e(k-1)$ is the error in speed in the last sample.

## 5 Simulink Diagram

In order to simulate data for the motor-generator system, a Simulink diagram had to be made that described both the system itself and the controller used to control the speed of the motor based on the error of $e(t) = \omega_d - \omega(t)$. Using the system equations described in the lab specification document[2], the blocks were able to be formed for the diagrams below. There were two different Simulink diagrams used for this experiment, the first had no load attached for Part A of the experiment, and the second had a load attached that varied according to a sequence determined by student number for Part B.

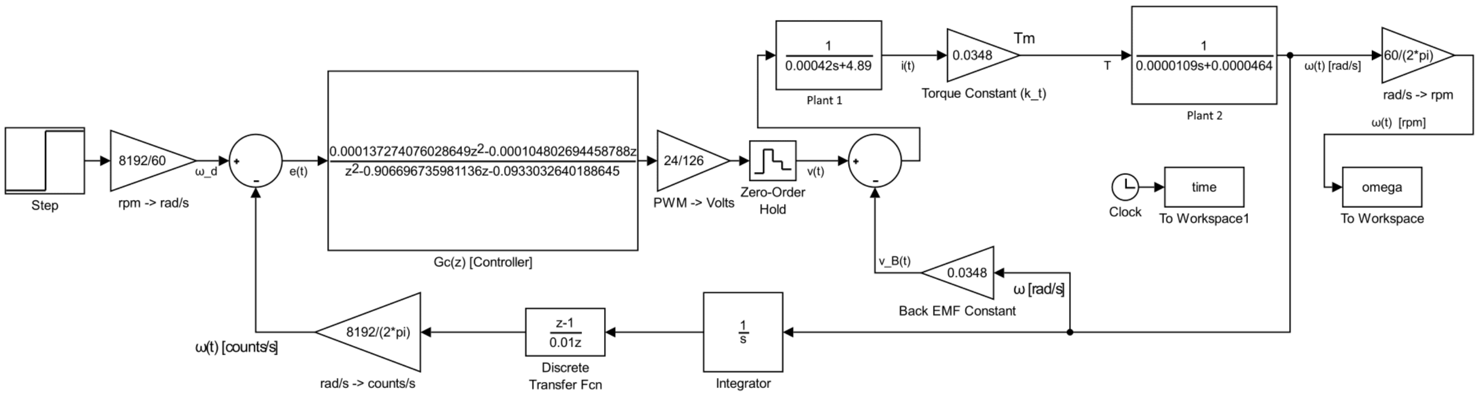

Figure 3: Simulink diagram of motor-generator system and discrete controller - no load attached

For Part B of the experiment, a varying number of resistors were connected to the output of the generator in parallel and hence generated a load that varied every few seconds. This can be seen in the ”Number of Resistors” block in the diagram below, it is a repeating stair input that will change the number of resistors used every 2 seconds (as found by inspection of experimental data).

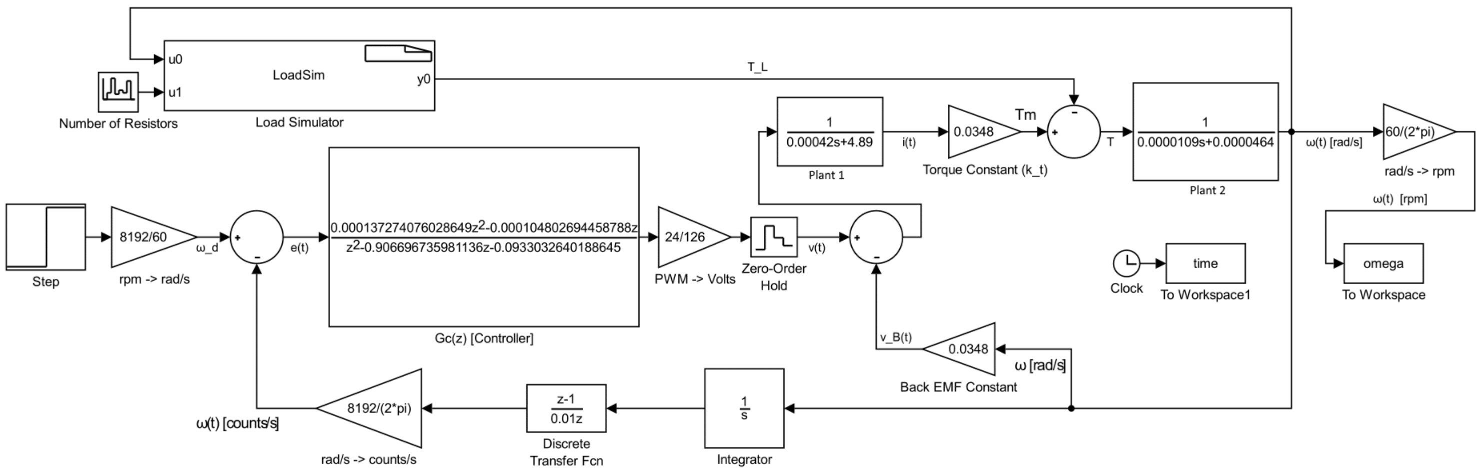

Figure 4: Simulink diagram of motor-generator system and discrete controller - varying load attached

Note that the step input simply represents an input of $\omega_d$ into the system, the number of resistors block is the number of resistors connected in parallel to the output of the generator, and the LoadSim block determines the load torque generated by the resistor circuit (See Lab Document Eq. (7), (8), and (9))[2].

The largest block in the above figure represents the discrete controller that was designed in the previous section using Eq. (20), it takes in the error $e(t)$ and outputs a value that is fed into a PWM amplifier. This PWM amplifier then outputs the voltage, $V(z)$ (See Lab Document Eq. (3))[2], that is zero-order held. This voltage then has the back EMF subtracted from it, and is fed into Plant 1 which relates voltage to current (See Lab Document Eq. (4))[2] and outputs a current that is then converted into the motor torque $T_m$ (See Lab Document Eq. (5))[2]. This torque is subtracted by the load torque for Fig. 4 and fed into Plant 2, or just fed into Plant 2 for Fig. 3 without subtracting anything (See Lab Document Eq. (10))[2]. This plant relates the torque to the motor speed which is what was needed (See Lab Document Eq. (11))[2].

## 6 Results

### 6.1 Part A - No Load

For this part of the experiment, no resistors were connected to the motor-generator system (hence no load) and there was a single step input applied of 1000 rpm at $t \approx 1.35$s (for the experimental data). The motor was initially run at 1000 rpm, and after the step input was applied, rose to a total 2000 rpm. Running the simulation given by the Simulink diagram, plotting it and superimposing this onto the plotted (and shifted to relevant section) experimental data, the following is obtained:

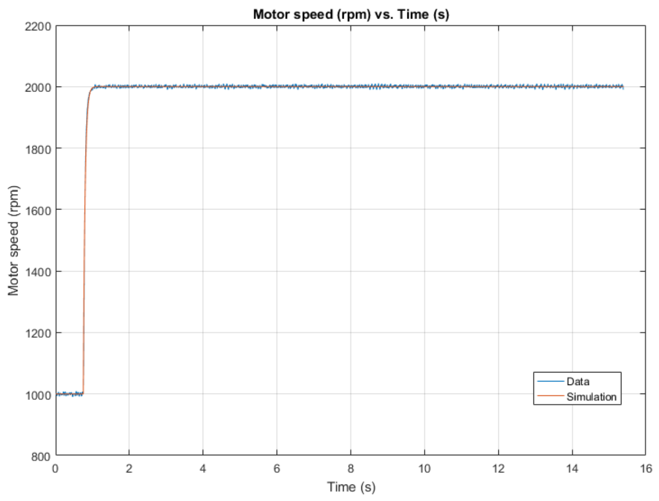

Figure 5: Simulink simulation against actual data of step input cycle

It can be seen from the above figure that, ignoring noise from the experimental data, the simulated plot is almost exactly the same as the experimental data. So it can be said that there is essentially zero steady-state error. However, for the design to be correct, the first-order response when the step input is applied has to have a time constant $\tau = \tau_d = 46$ms (taken from individual design data). Taking the relevant simulation data corresponding to the first-order response, zeroing it and curve-fitting using MATLAB's *cftool* and a fit equation of $f(t) = A(1 - e^{-t/\tau})$, the following is found:

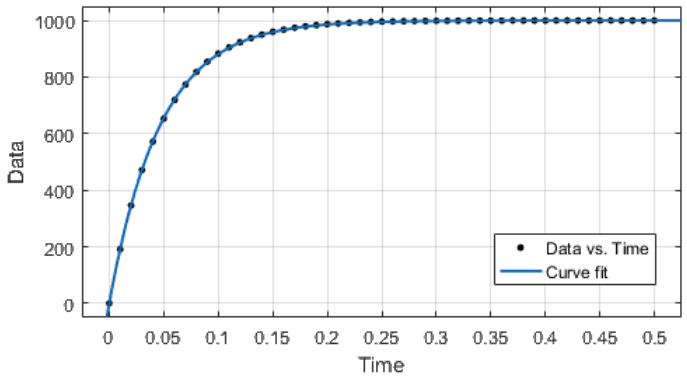

Figure 6: MATLAB *cftool* curve-fit to first-order response of simulated data

The above figure gives a good fit, and the resulting time constant, $\tau$, was 0.4699. Calculating the percentage error:

$$PE = \frac{|0.4699 - 0.46|}{0.46} \text{ x } 100\% = 2.15\% \tag{22}$$

This is a relatively small percentage error and falls within a margin that could be deemed to be accurate. Therefore, it can be seen that the design is correct and the controller is valid based on the above comparisons.

## 6.2 Part B - Varied Load

As seen in the Experimental Procedure section above, for this part of the experiment, a sequence of load resistors was applied to the output of the generator determined by student number. The particular sequence used in this report as noted above is:

$$0\rightarrow4\rightarrow13\rightarrow3\rightarrow5\rightarrow4\rightarrow11\rightarrow0$$

It was found from the supplied experimental data that the resistor loads were changed every $\approx$ 2s, determined by inspection (and so potentially subject to error).

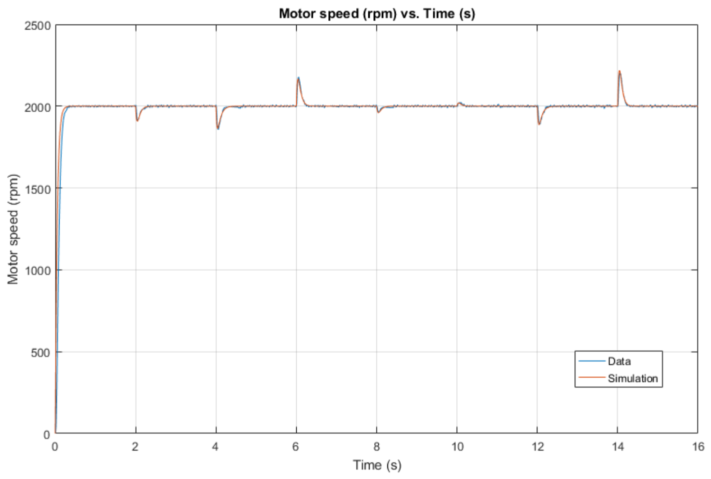

Figure 7: Simulink simulation vs. actual data with resistor load sequence

From the above, ignoring the noise in the experimental data it can be seen that the simulated data lines up with it very accurately. There is an exception with the transient period after the step input is applied, however this can be attributed to experimental errors such as the sensor not being accurate enough, it can be seen that the experimental data doesn't start at 0 initially and drops to 0 before then increasing from the plot below:

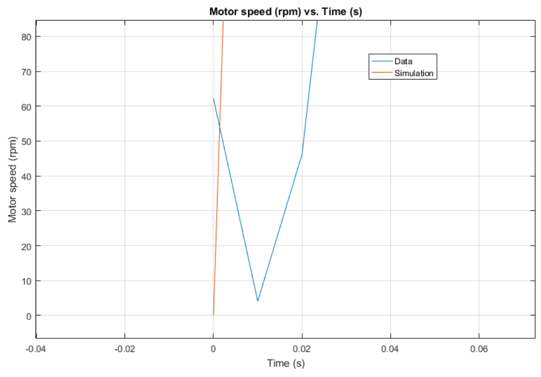

Figure 8: Magnification of above figure

It is possible that this error was amplified due to not modelling friction of the motor, however the friction is negligible and would most likely have no noticeable consequences.

## 7 Conclusion

Overall, the experiment was a success and a controller was able to be designed that gave a zero steady-state error to the desired motor speed, and was able to deal with disturbances in the load on the output of the generator. There were some discrepancies between the simulated and experimental data in the results of Part B, however it was determined that this was due to experimental error, and thus does not invalidate the controller design. This reasoning was further justified by the validation of the design shown in the results of Part A, with the simulated and experimental data lining up almost perfectly, and the time constant of the first-order response of the simulation having a discrepancy of only 2.15% to the experimental value. This small of an error was likely to have been caused by experimental errors such as the sensor not being accurate enough (which explains the noise in the data), just as was concluded with Part B.

## References

[1] Katupitiya, J. (2017). Laboratory Experiment II : Design of a Speed Controller. [pdf] p.1.
Available at:
https://moodle.telt.unsw.edu.au/pluginfile.php/2728239/mod_folder/content/0/SpeedControlExperiment2017.pdf
[Accessed Oct. 2017].

[2] Katupitiya, J. (2017). Laboratory Experiment II : Design of a Speed Controller. [pdf] p.3-4.
https://moodle.telt.unsw.edu.au/pluginfile.php/2728239/mod_folder/content/0/PostSpeedControlExperiment2017.pdf
[Accessed Oct. 2017].