DC Motors Studio Report
CG1111A Studio 7

Prannaya Gupta (B02)

September 14, 2022

Activity 1

Simulation

My last two digits are "67", thus as per the given formula, by Parameter Set Number is **18**.

Motor Voltage V_m (V)	Rotational Speed N (RPM)
7	691
8	1043
9	1354
10	1666
11	1981
12	2450

Table 1.1: Simulated Readings

For this studio, we note that $\omega = \frac{2\pi}{60}N$, thus the equation $\omega = \frac{V_m}{K_e} - \frac{R_m I_m}{K_e}$ is really just a linear line. Therefore, we use the linear trendline.

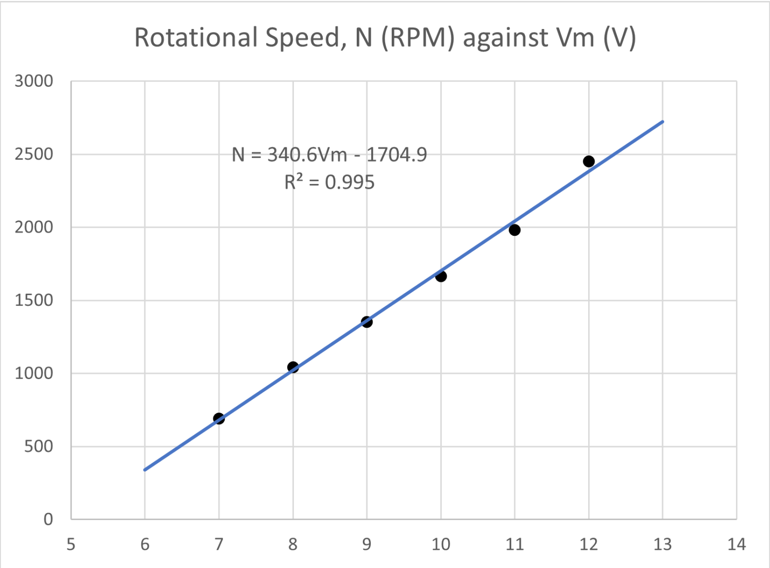

Figure 1.1: Graph of Rotational Speed, N (RPM) against Motor Voltage, V_m

From here, we get that the equation of the curve is:

$$N = 340.6V_m - 1704.9$$

The motor does not start turning immediately after the voltage is applied because it has to overcome the static friction which it is unable to immediately due to low torque.

Activity 2

DC Motor Characterization

Torque Load	Motor Current I_m (A)	Rotational Speed N (RPM)	ω (rad/s)
~2/3 of slider	0.769	861	90.16
~1/2 of slider	0.581	1530	160.22
~1/3 of slider	0.410	2450	256.56
~1/6 of slider	0.204	3212	336.36
~1/10 of slider	0.128	3444	360.65

Table 2.1: Simulated Readings

The following represents the graph plotted:

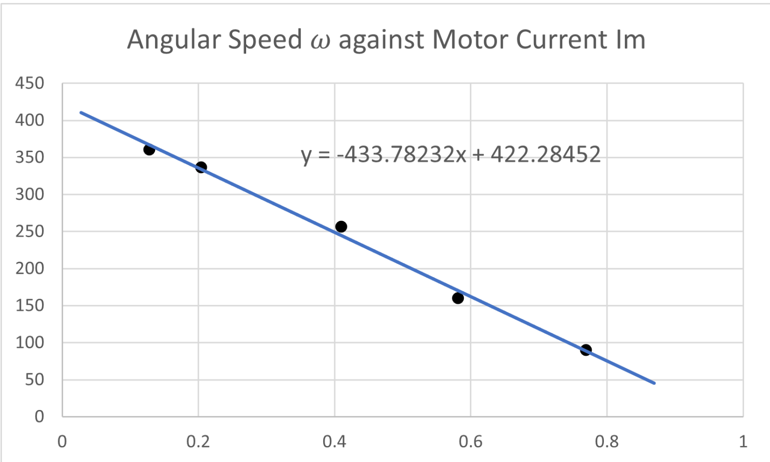

Figure 2.1: Graph of Angular Speed, ω (rad/s) against Motor Current, I_m

The equation of the curve is:

$$I_m = -0.00229 \times \omega + 0.97041$$

In stall conditions, $\omega = 0$, thus we have that:

$$I_{\text{stall}} = 0.97041 \approx \mathbf{0.970\ A}$$

In no-load conditions, $I_m = 0$, thus we have that:

$$\begin{aligned}
\omega_{\text{no-load}} &= \frac{0.97041}{0.00229} \\
\omega_{\text{no-load}} &= 423.75983 \text{ rad/s} \\
N_{\text{no-load}} &= 423.75983 \times \frac{60}{2\pi} \\
&= 4046.60830 \text{ RPM} \\
&\approx \mathbf{4050\ RPM}
\end{aligned}$$

As for the motor resistance, we can compute it under stall conditions as follows:

$$\begin{aligned}
V_m &= I_{\text{stall}} R \\
R &= \frac{12.0}{0.97041} \\
&= 12.36591\ \Omega \\
&\approx \mathbf{12.366\ \Omega}
\end{aligned}$$

For the back emf constant K_e, we can compute it under no-load conditions as follows:

$$\begin{aligned}
\omega &= \frac{V_m}{K_e} \\
K_e &= \frac{12.0}{423.75983} \\
&= 0.028318 \text{ V-s/rad} \\
&\approx \mathbf{28.318\ mV\text{-}s/rad}
\end{aligned}$$

Since this is a PMDC motor, we can simply say that $K_t = K_e = 0.028318$ V-s/rad $\approx$ **28.318 mV-s/rad**.

Activity 3

Actual Testing

PWM Frequecy (kHz)	t_p (in μs)	t_{on} (in μs)	PWM Noise Audible?
1	1000	500	Yes
2	500	250	Yes
4	250	125	Yes
10	100	50	No
20	50	25	No

Table 3.1: Audibility Based off Frequency

From your observations, which of the following frequencies in Table III would be more appropriate if there are humans working in the vicinity of a PWM-controlled motor?
20kHz. It's the least audible and will be the least annoying for such workers.

Duty Cycle (%)	t_{on} (in μs)
50	25
60	30
70	35
80	40
90	45
100	50

Table 3.2: Duty Cycle at PWM Frequency of 20 kHz

Can you observe the increase in speed as the duty cycle increases?
Yes, I noticed. It is very cool.

Activity 4

Challenge Yourself

Modify the Arduino code, so that the wheel will spin in one direction for 10 seconds, and then reverse for 10 seconds. Repeat this indefinitely. You may choose any duty cycle.

I am using Platform.io, hence the inclusion of the `include` line.

```
#include "Arduino.h"

// Give names to the pins of Arduino to be used and
// define their values as integer
int Ena = 7;  // IO port 7 will be connected to Pin 9 (Enable) of
   L293D
int Mot1 = 5; // IO port 5 will be connected to Pin 15 of L293D
int Mot2 = 6; // IO port 6 will be connected to Pin 10 of L293D
int tp;       // Define a variable for period of PWM
int tON;      // Define a variable for ON time
int tOFF;     // Define a variable for OFF time
long tTotal;

// Following section will be run once at the beginning.
void setup()
{
  pinMode(Ena, OUTPUT);     // OUTPUT from Arduino
  pinMode(Mot1, OUTPUT);    // OUTPUT from Arduino
  pinMode(Mot2, OUTPUT);    // OUTPUT from Arduino
  digitalWrite(Ena, HIGH); // Enable pin is set to logic HIGH
  digitalWrite(Mot1, LOW); // Both control pins are initialized
  digitalWrite(Mot2, LOW); // to LOW so motor won't spin for now
  tp = 1000;                // Period is 500 microseconds; must be
   >= tON
  tON = 500;               // ON for 300 microseconds
  tOFF = tp - tON;         // OFF for remaining time of the period
```

```
  tTotal = 10000000;
}

void loop()
{
  for(int i = 0; i < (tTotal/tp); i++) {
    digitalWrite(Mot1, HIGH);
    delayMicroseconds(tON);
    digitalWrite(Mot1, LOW);
    delayMicroseconds(tOFF);
  }
  for (int i = 0; i < (tTotal / tp); i++)
  {
    digitalWrite(Mot2, HIGH);
    delayMicroseconds(tON);
    digitalWrite(Mot2, LOW);
    delayMicroseconds(tOFF);
  }
}
```