Robotics Lab 1 Report:
Motor Characteristics

Christian Prather

Purpose:

The purpose of this lab is to determine the characteristics of the provided Lego motor so as to appropriately use them in future labs and calculations.

Methods:

Step 1)

The first step was to establish a mechanism that would allow for motor testing. This mechanism needed to be easy to reset, consistent, and low friction. The decision was made to allow for a single 1:1 ratio gear assembly mounted to the lab table. A single lightweight cup is attached to a string whose other end is mounted to the output gear of the system. This allows direct interpretation of motor characteristics without any conversion factor needed.

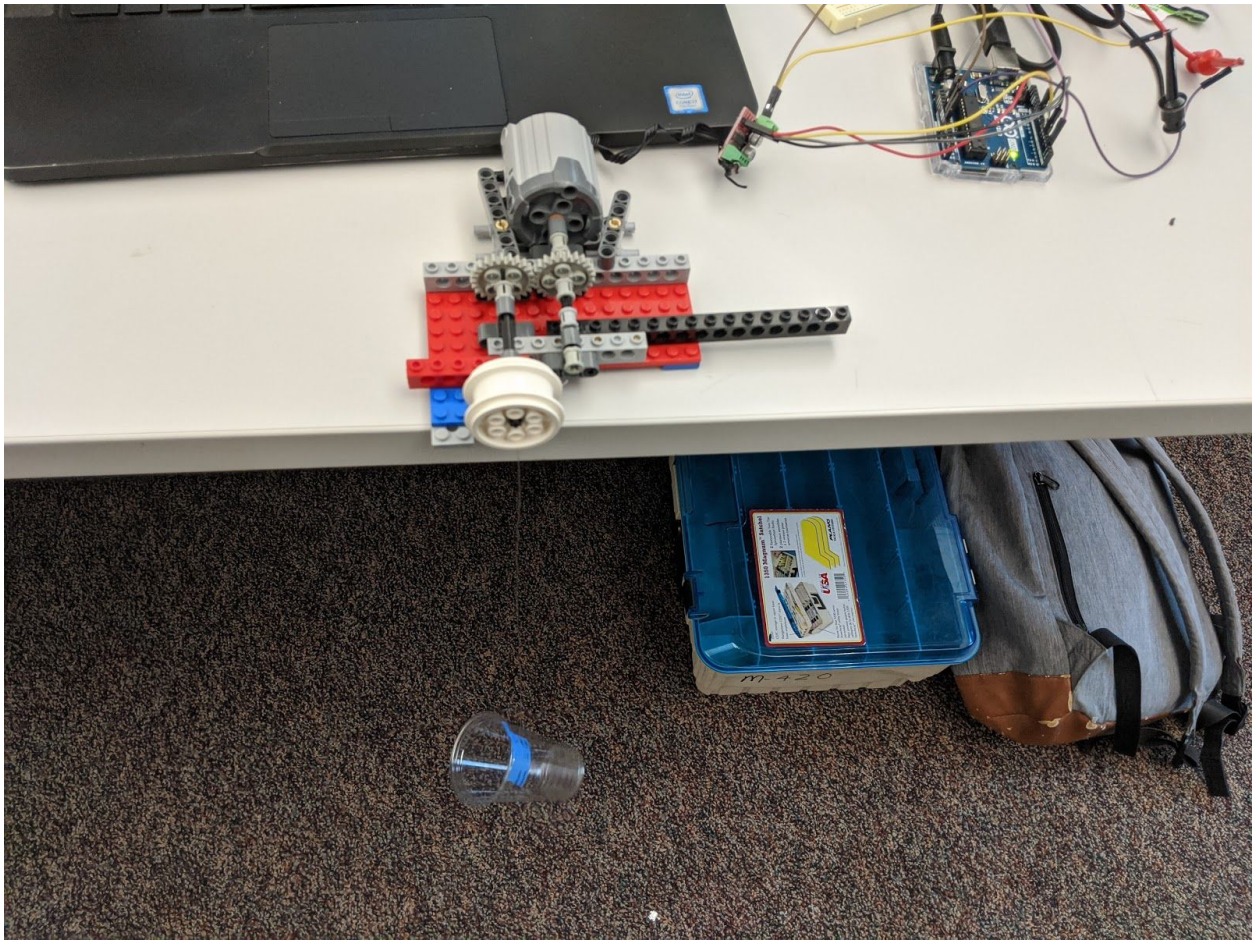
Step2)

The motor was wired to a simple H-bridge with an Arduino microcontroller and button utilized to start and stop its rotation. A simple testing procedure was followed of

- Add known weight
- Lift cup known constant distance
- Measure continuous current draw
- Recorde time taken

This was repeated until the motor hit stall torque.

The data was collected in a single spreadsheet and allowed for easy calculation and plotting. Torque was calculated with the equation M*9.81*r*1000 in Nmm.



Code:

The code was a simple program that allowed for single button control of the motor. Set up was done to configure the H-bridge pins to correctly map to those attached to the Arduino. The code then had two primary functions, initializing the motor and timing. A simple millis() call at the beginning and end of the button press allowed for a delta to be calculated giving us time the motor was active in milliseconds.

Results:

Motor 1:

Motor 1 Lifting Tests										
Number of Washers	Travel Time (s)	Current (A)	Speed (m/s)	w (rad/s)	w (rpm)	Mass (kg)	Torque (Nmm)	P_in (mW)	P_out (mW)	Efficiency (%)
0	2.26	0.160	0.235	11.741	112	0.000	0.000	1253	0	0.00
2	2.30	0.235	0.230	11.522	110	0.094	18.443	1840	212	11.55
4	2.22	0.280	0.239	11.937	114	0.188	36.886	2192	440	20.08
5	2.23	0.310	0.238	11.883	113	0.235	46.107	2427	548	22.57
6	2.34	0.360	0.227	11.344	108	0.282	55.328	2819	628	22.27
8	2.49	0.430	0.213	10.643	102	0.376	73.771	3367	785	23.32
10	2.78	0.520	0.191	9.532	91	0.470	92.214	4072	879	21.59
12	2.98	0.550	0.178	8.893	85	0.564	110.657	4307	984	22.85
14	3.15	0.670	0.168	8.413	80	0.658	129.100	5246	1086	20.70
16	3.74	0.700	0.142	7.086	68	0.752	147.542	5481	1045	19.07
18	3.87	0.710	0.137	6.848	65	0.846	165.985	5559	1137	20.44
20	5.86	0.735	0.090	4.522	43	0.940	184.428	5755	834	14.49
24			#DIV/0!	#DIV/0!	#DIV/0!	1.128	221.314	0	#DIV/0!	#DIV/0!
26			#DIV/0!	#DIV/0!	#DIV/0!	1.222	239.756	0	#DIV/0!	#DIV/0!

Fig 1

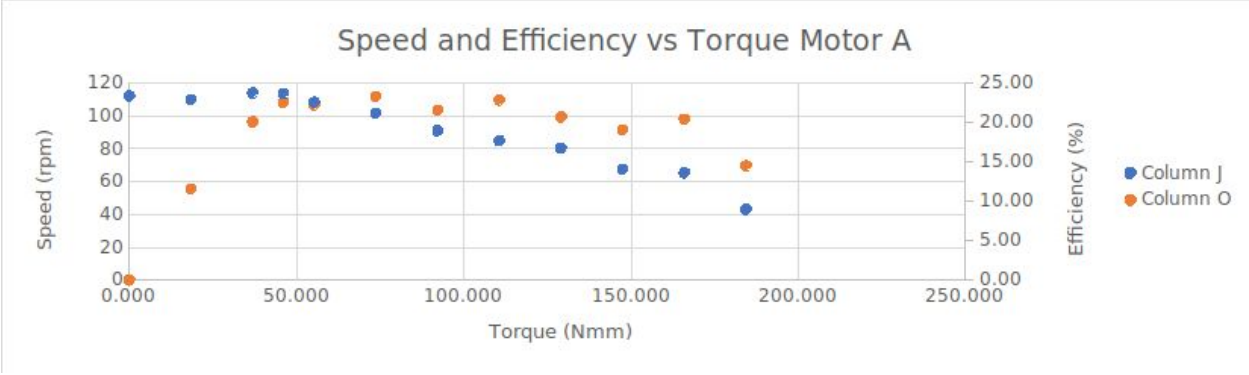


Fig 2

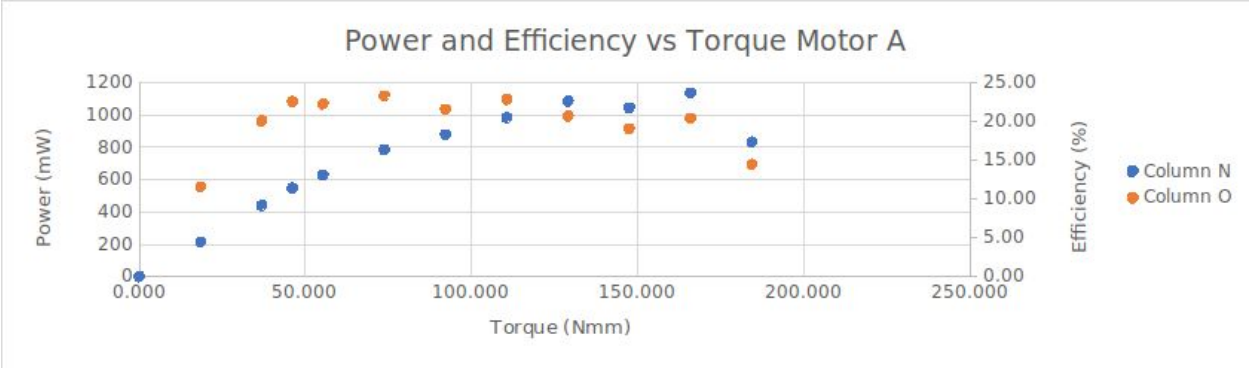

Power and Efficiency vs Torque Motor A
Power (mW)
Efficiency (%)
Torque (Nmm)
0.000
50.000
100.000
150.000
200.000
250.000
0
200
400
600
800
1000
1200
0.00
5.00
10.00
15.00
20.00
25.00
Column N
Column O

Fig 3

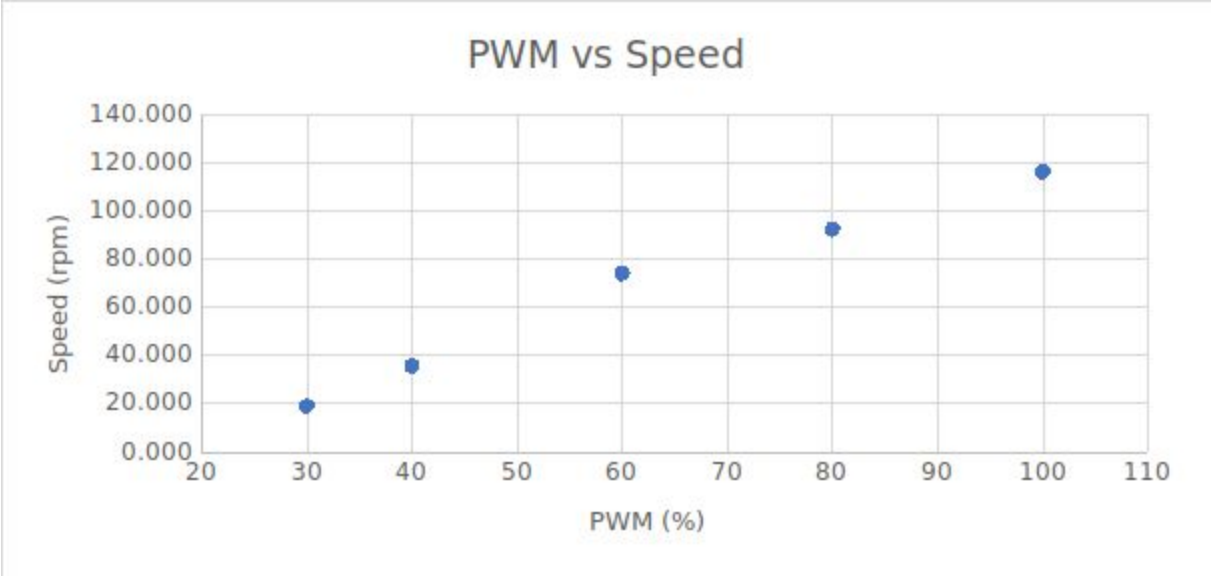

PWM vs Speed
Speed (rpm)
PWM (%)
0.000
20.000
40.000
60.000
80.000
100.000
120.000
140.000
20
30
40
50
60
70
80
90
100
110

Fig 4

Motor 2:

Motor 2										
Number of Washers	Travel Time (s)	Current (A)	Speed (m/s)	w (rad/s)	w (rpm)	Mass (kg)	Torque (Nmm)	P_in (mW)	P_out (mW)	Efficiency (%)
2	1.88	0.320	0.282	14.096	135	0.094	18.443	2506	260	10.38
4	2.33	0.360	0.227	11.373	109	0.188	36.886	2819	420	14.88
5	1.44	0.420	0.368	18.403	176	0.235	46.107	3289	848	25.80
6	2.66	0.450	0.199	9.962	95	0.282	55.328	3524	551	15.64
8	2.89	0.520	0.183	9.170	88	0.376	73.771	4072	676	16.61
10	3.12	0.570	0.170	8.494	81	0.470	92.214	4463	783	17.55
12	3.19	0.590	0.166	8.307	79	0.564	110.657	4620	919	19.90
14	3.61	0.683	0.147	7.341	70	0.658	129.100	5348	948	17.72
16	5.20	0.730	0.102	5.096	49	0.752	147.542	5716	752	13.15
18	5.25	0.770	0.101	5.048	48	0.846	165.985	6029	838	13.90
20	6.59	0.780	0.080	4.021	38	0.940	184.428	6107	742	12.14
24	STALL	0.900	#VALUE!	#VALUE!	#VALUE!	1.128	221.314	7047	#VALUE!	#VALUE!

Fig 5

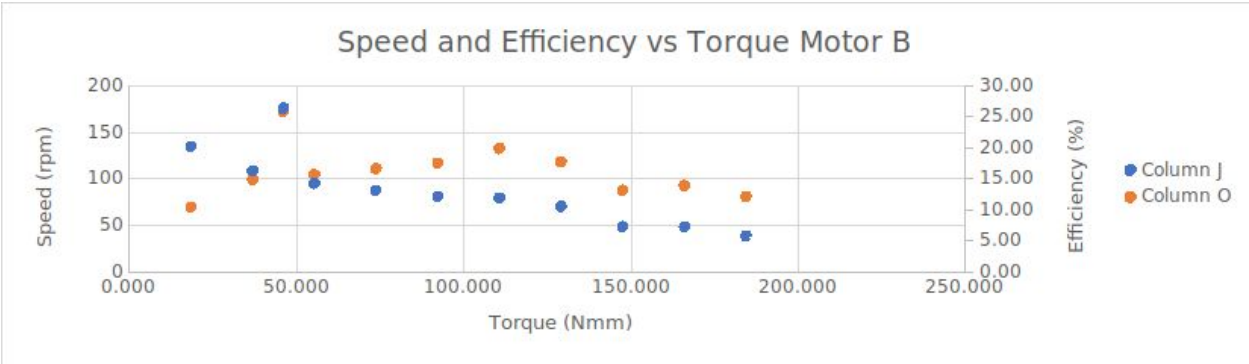


Fig 6

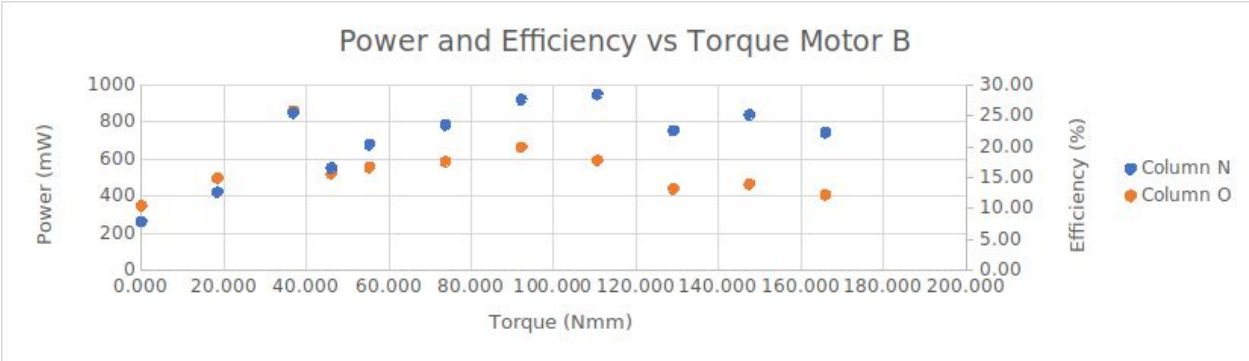


Fig 7

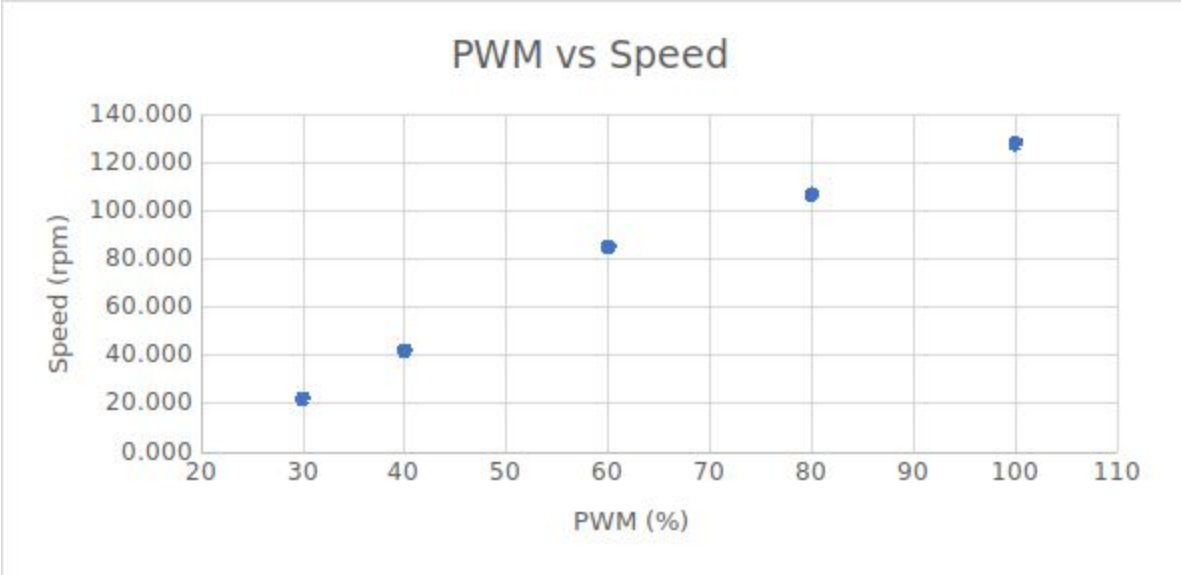


Fig 8

As can be seen in the above figures both motors follow fairly reasonable trends as would be predicted from standard DC brushed motor characteristics.

Conclusion:

The Speed Torque curves are very nice in terms of following a trend expected by DC brushed motors. It can be expected that a linear relation will exist among the motors. As Speed decreases the Torque should rise. This can be clearly seen in both figures (2&6) as the speed reduces the torque rises. You can also see from the same plots the quadratic relationship between efficiency and speed. As the motor increases speed its efficiency rises until it maximizes at the no load rpm /2 speed at which point as it gains speed the efficiency drops leaving a quadratic relationship.

By looking at the figures shown to address the power/torque/efficiency relations (3 & 7) you can clearly see that Power and Efficiency follow each other closely in a quadratic relationship with torque. That is peak efficiency happens as the power hits its peak. This makes sense intuitively as you would expect the best power output with the best system efficiency. You do however see that a rise in torque past this maximum yields to a direct drop in both efficiency and power output.

The PWM and speed (Fig 4 & 8) shows a very linear trend line that is as PWM increases the duty cycle closer to 100% you see higher speed in rpm.

The stall torque for both motors was found to be at 21 washers ~0.987 kg. This indicated that if I were to design my robot around peak efficiency with no gearing I should ensure I do not exceed a weight of 0.987 kg as each motor can handle half. The actually stall torque value being ~ 220 Nmm.

References:

Class Notes in Mines MEGN 441

Appendices:

Constants	
Radius of Pulley Wheel (m):	0.020
Distance Travelled (m):	0.530
Mass per Washer (kg):	0.047
System Voltage (V):	7.83

1 Washer

Motor 1 PWM tests		
PWM setting (%)	Travel Time(s)	w (rpm)
30	13.39	18.895
40	7.10	35.642
60	3.42	73.993
80	2.74	92.356
100	2.18	116.081

Motor 2 PWM tests		
PWM setting (%)	Travel Time(s)	w (rpm)
30	11.49	22.028
40	6.04	41.905
60	2.98	84.934
80	2.37	106.795
100	1.98	127.830

Fig 1 -> Motor 1 Collection data
Fig 2 -> Motor 1 Speed v Efficiency v Torque Plot
Fig 3 -> Motor 1 Power v Efficency v Torque Plot
Fig 4 -> Motor 1 PWM v Speed Plot
Fig 5 -> Motor 2 Collection data
Fig 6 -> Motor 2 Speed v Efficiency v Torque Plot
Fig 7 -> Motor 2 Power v Efficency v Torque Plot
Fig 8 -> Motor 2 PWM v Speed Plot

***Flow*:**
Button Press -> millis() -> motor drive forward -> Button relese -> millins() -> delta calculation -> motor stop

Code provided for students in lab:

```
/* another version of Digital Read
*  mshapiro 0842020
*/

#define pushButton 2 // install a Pullup button with its output into Pin 2

// If you have a kit with the moto shield, set this to true
// If you have the Dual H-Bridge controller w/o the shield, set to false
#define SHIELD false

// Defining these allows us to use letters in place of binary when
// controlling our motor(s)
#define A 0
#define B 1

//SHIELD Pin varables - cannot be changed
#define motorApwm 3
#define motorAdir 12
#define motorBpwm 11
#define motorBdir 13

//Driver Pin variable - any 4 analog pins (marked with ~ on your board)
#define IN1 5
#define IN2 6
#define IN3 9
#define IN4 10

#define ledPin 13 //LED is connected to Pin 13
```

```
boolean buttonPushed = false; // set initial button variable to 0/false/Off

//variables used in loop
int buttonState = 0;

// time is being measured in milliseconds, which makes the numbers large
// the long variable type allows us to store/use big numbers
long starttime, stoptime, deltatime;



// the setup routine runs once when you press reset:
void setup() {

  // set the pushbutton's pin as a pull-up input:
  pinMode(pushButton, INPUT_PULLUP);

  //set the LED pin as an output
  pinMode(ledPin, OUTPUT);

  //use pre-built functions for motor setup (found in 2nd tab above)
  motor_setup();

  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);  // set the serial comm to 9600 BAUD
}

// the loop routine runs over and over again forever (or until reset button/loss of power):
void loop() {

  // read the input pin:
  buttonState = digitalRead(pushButton);

  if(buttonState == 0){ // If button is pushed
    run_motor(B, 255 * 1.0); // motor A is on at full speed (pwm = 0-255)
    digitalWrite(ledPin, HIGH); //turn LED on
    if (buttonPushed == false){ // we are coming out of the "OFF" state
      buttonPushed = true; // set new "ON" state
      starttime = millis(); // start timer
      Serial.print(starttime); // print start time
    }
  }
```

```
  else { // if button is not pushed
    run_motor(B, 0); //t urn motor off
    digitalWrite(ledPin, LOW); // turn LED off
    if (buttonPushed ==true) { // we are coming out of the "ON" state
      stoptime = millis(); //stop timer
      Serial.print(" ");
      Serial.print(stoptime); //print stop time
      buttonPushed = false; // set new "OFF" state
      deltatime = stoptime - starttime; // find time differnce
      //print results
      Serial.print("\t");
      Serial.println(deltatime);
    }
  }
  // Useful to debug button hardware:
  // print out the state of the button:
  //Serial.println(buttonState);
  delay(10);        // delay 10 ms in between reads for stability
}
```