STUDENT NAME AND ID: Alexandre BENOIT/199209819

COURSE WORK 2 – Robotic Project

SUMISSION DATE 13/12/2019

INTRODUCTION

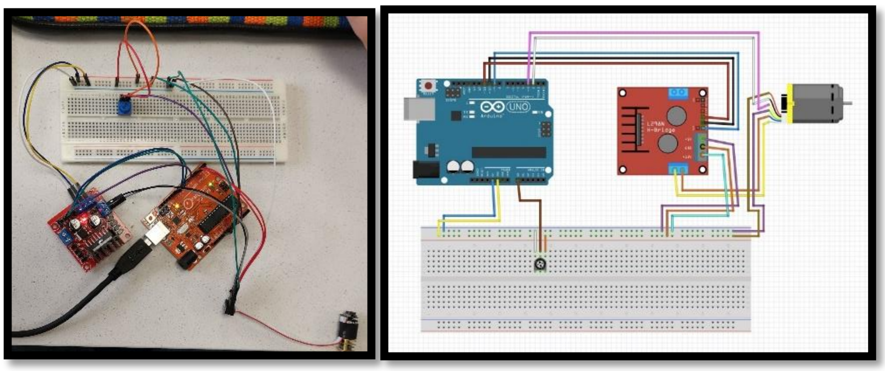
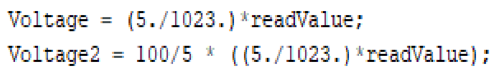


Figure 1- Circuit set-up

An experiment was done using several components as **Micro metal gear dc motor** equipped with a magnetic encoder (which measures rotation and allows us to control his speed and position control). An H-Bridge is also equipped. It is an electrical circuit that switches the polarity of voltage of the motor in order to control his rotation. It forms like an H-like configuration and we can change the current direction which will change the motor's rotation direction in order to control voltage difference and motor direction/speed. They are all implemented to a circuit and tested thanks to an Arduino.
The objective of this project is coding, thanks to Arduino's platform, a program which measures variation of voltage and control a dc motor.
It is an interesting project which give us the tools to manufacture spinning devices as the wheels of a car or thanks to exercise e, eventually control the rotation of a robot arm.

I. DESCRIPTION OF THE WORK



```
Voltage = (5./1023.)*readValue;
Voltage2 = 100/5 * ((5./1023.)*readValue);
```

Figure 2- exercise A important formula

Exercise A: We needed to measure the voltage variation on the variable resistor (trimmer). In generating an ARDUINO code (part of it **Figure 1**), we can show the values in Voltage as well as the corresponding percentage. In rotating the trimmer, we can change his value 0V to 5V which correspond respectively to 0% and 100% (formula **Figure 2**). It will be useful to control voltage difference as 5V will become our maximum speed.

Exercise B: We integrated two more devices: H-Bridge and the DC motor. H-Bridge and DC motor are connected via two wires to M1 and M2 which are two motor inputs. It allows the DC motor to turn thanks to the code line: **digitalWrite(in1Pin,LOW), digitalWrite(in2Pin,HIGH).**

Exercice C and D: We had to download the encoder library which will allow Arduino platform to read encoder's values and signal. The aim here was manipulating the DC motor's direction (forward/backward in pressing + or -) and evaluating his speed with the trimmer. In fact, 0V is no speed and the more volts you have the faster the rotation of the motor goes. In order to do that, we added a **millis()** function which will record the new/old time. It is an important function as it will give us the RPM thanks to this equation: **(-(newposition-oldposition)/12)/((newtime-oldtime)/1000)**; **newposition** is given by the encoder.

Exercise E: We improved the previous programs to turn the DC motor + encoder into a servo motor controlled by the trimmer. A servomotor is a rotary actuator with which we can controlled the angular position of the motor. The circuit is still the same, but the difficulty of this exercise was based on the code. The error was the problem to sort out and the given formula **errorPrev = error** which stores error as previous error for next loop that we are going to compare. **error = angle - trimmerDeg;** this line will calculate the difference between the desired and real angle. Finally, **pwm = (abs(kp*error))+(kd*(errorPrev-error)/(timeDif));** as we have seen during lecture, it gives PID formula where errorPrev-error/timeDif, gives the gradient of the error function and hence the derivative.

II. DESCRIPTION OF RESULTS:

Exercice A: As we turn the trimmer, voltage difference is changing. In pressing serial monitor and tracer command, it appears the voltage and its corresponding percentage (**Figure 3a)**. The graph is the variation of the voltage difference while I am turning the trimmer (**Figure 3b).**

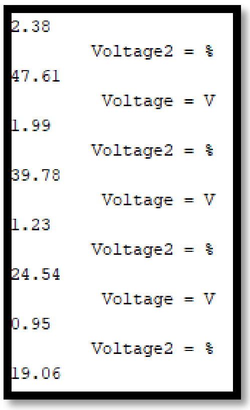
Exercice B: We were only able to turn the motor forward and backward to a constant speed in pressing + (forward), - (backward) or = (halt) in serial monitor. In calculating the voltage across the voltage wires with the voltmeter, we know that the rpm of the motor with 3.36V (and for 6V → 420 RPM) is 235.

Exercice C and D: Regarding exercises C and D, serial monitor can show the direction, the speed (RPM) (**Figure 4)** and its percentage. Moreover, the initial value of the DC motor makes it lag. That is why we introduced a reducing factor on speed, **speed1 = speed1 *0.25**, which makes his speed more accurate. To calculate the speed, we needed to take into account that the encoder reads 12 counts per revolution. Furthermore, it's a quadratic magnetic encoder (3 slots* 4 =12 counts).

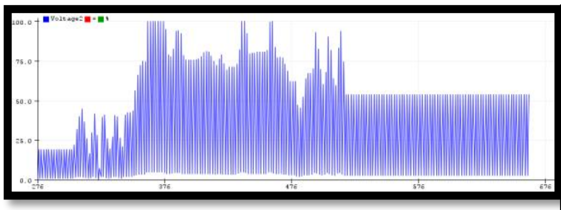
Exercice E: Now, the motor is turned into a servo motor where we can control his angle rotation with the trimmer. The **Figure 5** shows the position in degrees, the velocity in RPM, the PWM and trimmer degree in percentage. I had to play with kd and kp values in order to have an accurate servo motor otherwise I wasn't able to control it and it was turning on his own.

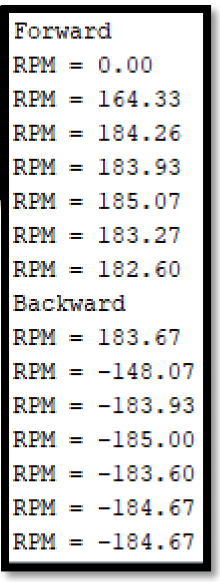
<u>CONCLUSION</u>

This project is a good understanding of how to program and run a DC motor which is the base of all the robotic in our days. On one hand, we learned how to control the speed and direction with the trimmer and showing speed value in RPM (forward/backward/halt) of a DC motor. On the other hand, we now know how to control a servo motor controlled by a trimmer which gives the ability to turn the motor with a certain degree angle.
Despite all of that, it is interesting to be confronted to a real-life project where we are supposed to apply our theorical knowledge and face practical issues as short-circuit, burned wires or circuit building problems. As I broke my H-Bridge in plugging my Arduino to a generator while changing my circuit, I realized as a future engineer that we have to be extremely careful with manipulation as mistake can cost a lot of money.



```
2.38
        Voltage2 = %
47.61
         Voltage = V
1.99
        Voltage2 = %
39.78
         Voltage = V
1.23
        Voltage2 = %
24.54
         Voltage = V
0.95
        Voltage2 = %
19.06
```



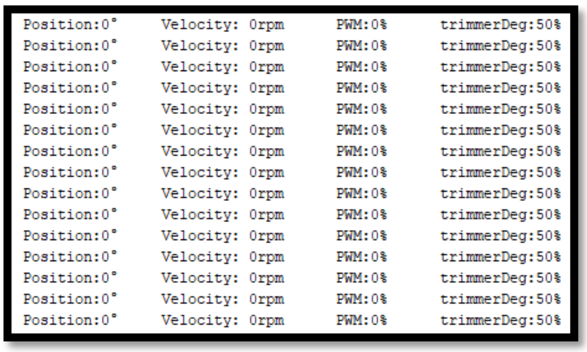


```
Forward
RPM = 0.00
RPM = 164.33
RPM = 184.26
RPM = 183.93
RPM = 185.07
RPM = 183.27
RPM = 182.60
Backward
RPM = 183.67
RPM = -148.07
RPM = -183.93
RPM = -185.00
RPM = -183.60
RPM = -184.67
RPM = -184.67
```

<u>Images database</u>

Figure 4- RPM values

Figure 3a and b- serial monitor and tracer



```
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
Position:0°    Velocity: 0rpm    PWM:0%    trimmerDeg:50%
```

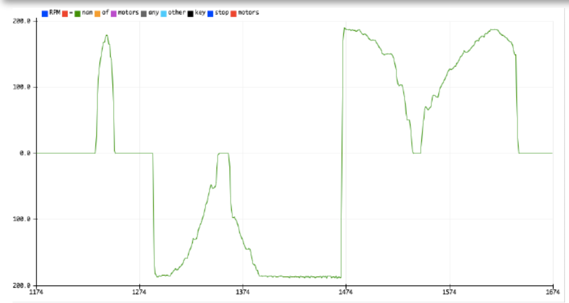


Figure 5- serial monitor

Figure 6- serial tracer