*Romanian Journal of Petroleum & Gas Technology*
*VOL. V (LXXVI) • No. 2/2024*

# DC MOTOR ANGULAR POSITION CONTROL SYSTEM USING ARDUINO PLATFORM

**Dragos-Ioan Matei** [1]

**Alexandru Negroiu** [1]

**Gabriela Bucur** [1*]

[1] Petroleum-Gas University of Ploiesti, Romania
[*] e-mail: gbucur@upg-ploiesti.ro

**DOI: 10.51865/JPGT.2024.02.07**

## ABSTRACT

The objective of this paper is to realize an angular position control system with a direct current motor. For this purpose, an Arduino microcontroller board is used to control the LS-25GA370 DC motor. The L298N dual H-bridge motor driver is used to modulate the PWM signal and control the travel direction. PID (proportional, integral and derivative) control algorithms are used to generate the PWM output. To implement a PID controller, the tuning parameters of the controller, namely proportional gain, integral gain and derivative gain must be carefully determined. Since the system is nonlinear, the PID controller is the best choice to accomplish this task. The sensor used was a Hall type encoder. To test the system, a LabVIEW interface was used, which communicates with the Arduino Uno platform through the serial port. This technical solution, using PID controller, can be applied for robotic arm position control system and many other industrial applications.

**Keywords:** DC motor, angular position, PID Controller, Arduino Uno, LabVIEW

## INTRODUCTION

The term "DC motor" is used to refer to any rotating electrical mechanism that transform direct current energy into mechanical energy. The DC motors is one of the most used motor, starting from small ***engines,*** used in electric toys and home applications, to complicated mechanisms for vehicles, elevators.

The DC motors include two components: a stator (inductor) and a rotor (armature). The stator is the stationary part of a motor, while the rotor is the rotating part. The stator is made of a cast iron or steel casing in the core of which the poles (main and auxiliary) are fixed with the respective windings (inductors) or without windings in the case of permanent magnets [6, 8, 12]. The ***rotor*** is made of electrotechnical steel plates, fixed on the shaft, having peripheral notches where the active sides of the inductor coils are located. The rotation of the rotor induces a voltage in the rotor winding. This induced voltage is opposite to the voltage applied to the rotor. As the motor rotates faster, the resulting voltage is almost equal to the induced voltage. The motor speed will remain constant as long as no load is acting on the motor. When a load is applied to the rotor, the

voltage will be reduced, but the motor will draw more current to do mechanical work [13-16].The most used types of DC motors are brushed and brushless. ***Brush motors*** are used for applications where the control system is simple, such as those for basic industrial equipment [2, 3, 5, 8, 10, 22, 26].

The most widespread DC motor driving method is PWM or Pulse Width Modulation, which works by alternating the applied voltage from high values to low values and vice-versa [4]. All most the researches presented of the literature focuses on controlling DC motors with simple feedback loops, using tunable compensators. The new and modern researches investigates advanced tuning techniques relating to artificial intelligence and optimization algorithms. For the present work, a DC motor with brushes was used.

DC motors, due to their simplicity in operation, ease of application, reliability and cost-effectiveness, are used in various applications, such as the defense industry [17, 19], the chemical industry [7], industrial robotics [1, 18]. In general, for a DC motor, speed control can be controlled by varying the terminal voltage. Modify the position of a DC motor is very important in various applications [20-23], this is done by taking a signal representing the required angle and driving the motor to that position. Devices that can provide easy control of a DC motor are microcontrollers.

The scope of this paper is to design and realize a system to control the angular position with a DC motor. To achieve this goal, the authors took into consideration to use an Arduino Uno assembly based on L298N bridge as driver. The sensor used was a Hall type encoder. To test the system, a LabVIEW interface was used, which communicates with the Arduino Uno platform through the serial port.

## SYSTEM DESIGN

Figure 1 shows the block diagram of the angular displacement control system with a direct current motor, after which the components and the role of each component in the system are detailed. The control element of the system is the Arduino development board, the sensor is represented by a Hall type encoder and the function of the execution element is performed by an H-bridge (L298N driver), to control the motor speed through PWM signals. The computer is used both to program the Arduino board and to support the LabView operator interface.

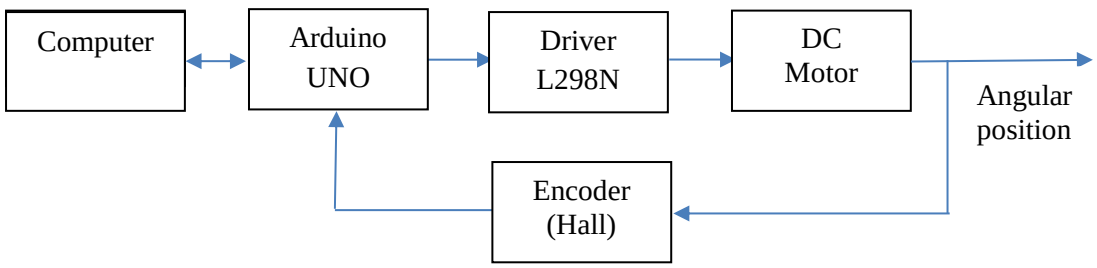


***Figure 1***. *Block diagram of the angular displacement control system with DC motor*

The LS-25GA370 DC motor is used for this project. It incorporates a reducer of up to 200 revolutions per minute. This type of motor works very well with the Arduino development board. The motor is also equipped with a 2-channel encoder, represented by two Hall-type sensors, arranged asymmetrically.

***Figure 2***. *LS-25GA370 direct current motor and built-in encoder* [12]

The Arduino Uno development board (Figure 3), the system controller, is made based on the 8-bit ATmega328P microcontroller [10]. To program the Arduino Uno development board, it is necessary to use the Arduino IDE (Integrated Development Environment) programming environment.

The execution element of the control system is represented by the L298N driver. This is actually a high-current dual H-bridge designed to work with standard TTL logic levels to drive inductive loads such as relays, electromagnets, DC motors, and stepper motors (Figure 4). This driver is compatible with Arduino.

The microcontroller communicates through dedicated I/O pins and through PWM, allowing to control the position and speed of the DC motor. It is also provided with an integrated 5V regulator, so if the supply voltage is up to 12V, it is not necessary to supply the logic part separately. In order to dissipate all the heat that accumulates during use, the integrated circuit on the driver is equipped with a radiator [10].

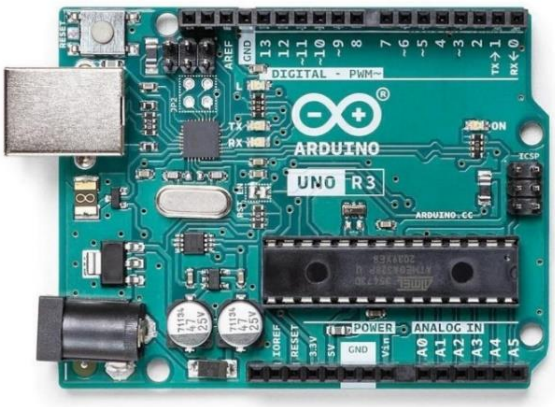


***Figure 3***. *Arduino Uno* [25]

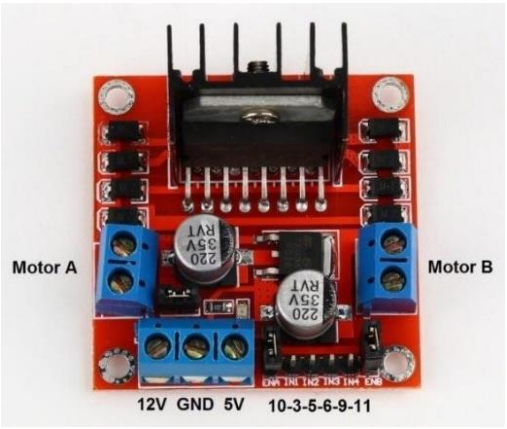


***Figure 4***. *Driver L298N* [24]

## HARDWARE AND SOFTWARE IMPLEMENTATION

The connection diagram between these elements is presented in Figure 5. It is important to specify that there is a USB (Universal Serial Bus) serial connection between the laptop and the Arduino development board. The completed assembly is presented in Figure 6.

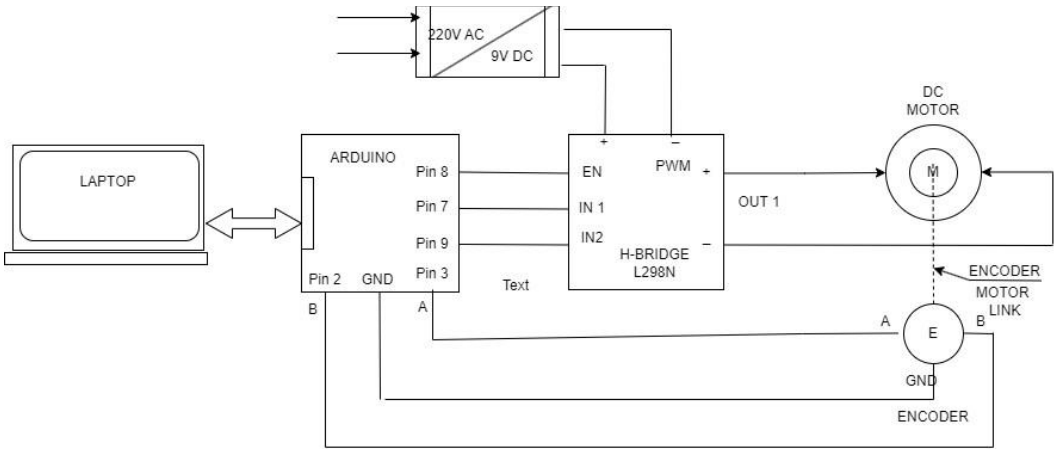


***Figure 5**. Electronics connection* [9]

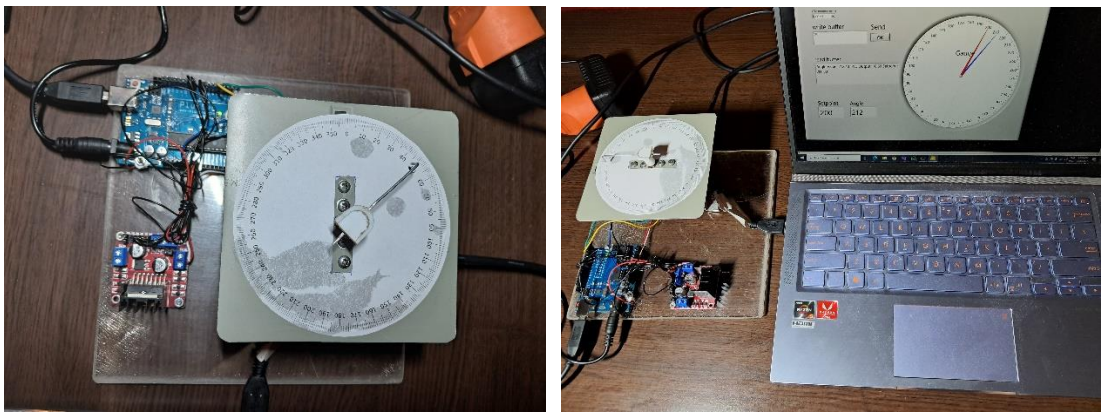

***Figure 6.** Implementation of the angular displacement control system* [9]

After completing the construction of the hardware assembly, in the programming environment of the Arduino Uno development board, the code for the interoperation of the components was created. The angular displacement control system with direct current motor uses the PID algorithm, and the tuning parameters will be found in the program symbolized $K_p$, $Ki$, $K_d$. The transfer function of the PID controller is of the form:

$$H_R(s) = K_P + K_i \cdot \frac{1}{s} + K_d \cdot s, \quad (1)$$

where $K_i = K_P \cdot \frac{1}{T_i}$ and $K_d = K_P \cdot T_d$. The proportional component has a variable $Kp$, also called amplification coefficient, which can reduce the response time, reduce the error but not eliminate it [11, 20]. Integral control is represented by the variable $Ki$, used to eliminate the steady-state error, but can worsen the transient response. The derivative

control uses the variable $K_d$ [s], used to increase the stability of the system and which influences the speed of increase or decrease of the error, having an anticipatory character, improving the quality of control. The logic diagram of the system's operation is presented in Figure 7.

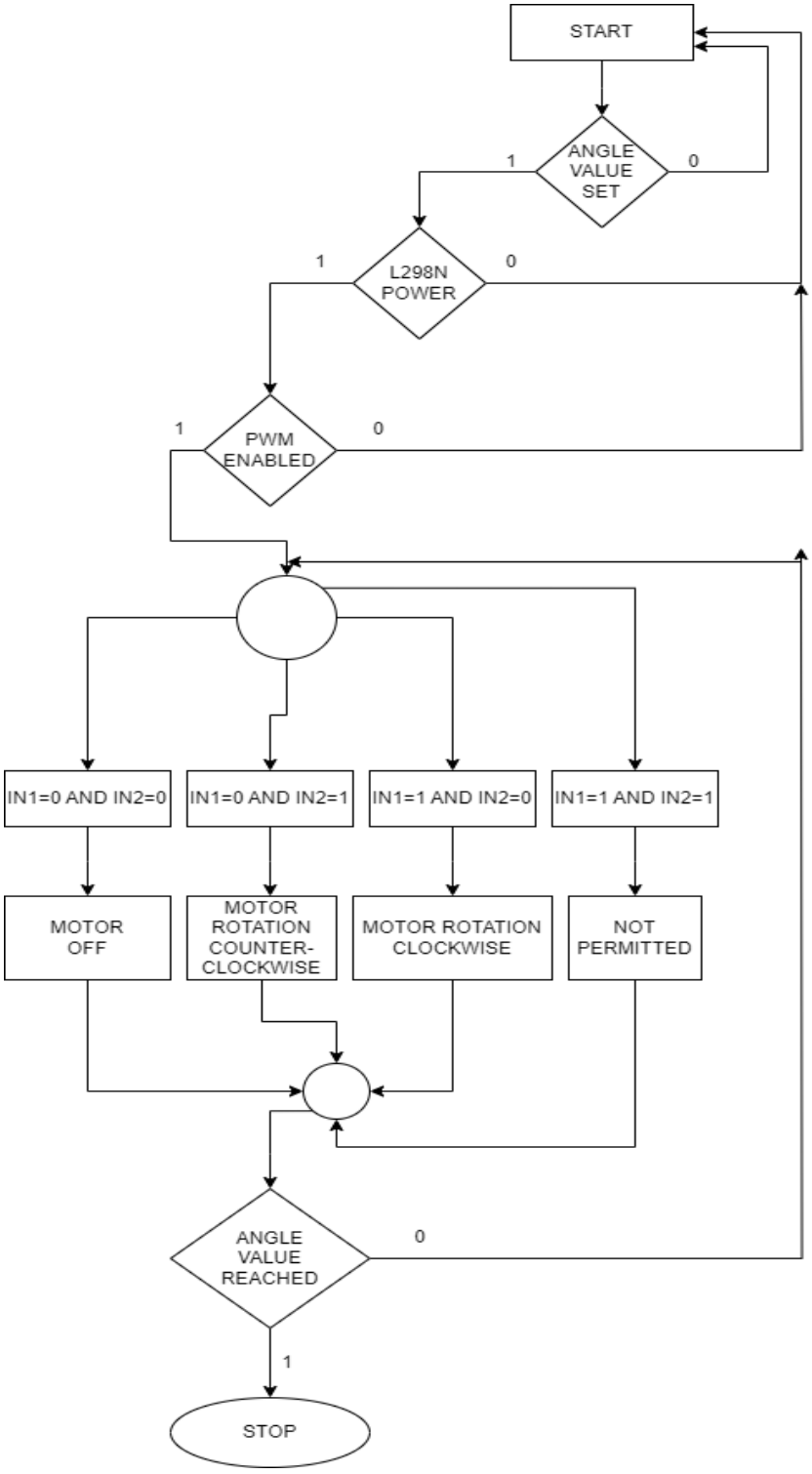


***Figure 7.*** *The logic diagram of the system's operation*

The program implemented in the microcontroller is the following:

```
#include <PID_v1.h>
#include <Encoder.h>
// Define the motor pins
int motorPin1 = 7;
int motorPin2 = 8;
int enablePin = 9;
// Define the encoder pins
Encoder myEncoder(3, 2);
// Define the PID variables
double Setpoint, Input, Output, Angle;
double Kp =0.11, Ki =0.08, Kd =0.05;
PID myPID(&Input, &Output, &Setpoint, Kp, Ki, Kd, DIRECT);
void setup()
{
// Set the motor pins as outputs
pinMode(enginePin1, OUTPUT);
pinMode(motorPin2, OUTPUT);
pinMode(enablePin, OUTPUT);
// Set the encoder pins as inputs
pinMode(3, INPUT_PULLUP);
pinMode(2, INPUT_PULLUP);
// Set the PID values
Setpoint = 0;
myPID.SetMode(AUTOMATIC);
myPID.SetOutputLimits(-255, 255);
// Begin serial communication
Serial.begin(9600);
}
void loop()
{
if (Serial.available() > 0)
{
Serial.println("Write a number to store in memory:");
Setpoint = Serial.parseInt();
Serial.print("You wrote: ");
Serial.println(Setpoint);
Serial.println("Reading from memory...");
Serial.println(Setpoint);
}
// Read the encoder value
Input = myEncoder.read()/2.5;
Angle=Input;
// Compute the PID output
myPID.Compute();
// Set the motor speed
```

```
if (abs(Input-Setpoint) > 1)
{ // If the difference between the desired and the current position is greater
than 1 pulse
if (Input < Setpoint)
{
digitalWrite(motorPin1, HIGH);
digitalWrite(motorPin2, LOW);
} if (Input > Setpoint)
{
digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, HIGH);
}
}
else
{
digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, LOW);
}
analogWrite(enablePin, abs(Output));
// Print the encoder value and PID output
Serial.print("Angle Value: ");
Serial.print(Angle);
Serial.print("PID Output: ");
Serial.print(Output);
Serial.print(" Setpoint: ");
Serial.println(Setpoint);
delay(100);
// Wait for a moment
}
```

## ANGULAR POSITION CONTROL SYSTEM TESTING

To verify the operation of the angular position control system, an application was created in the LabVIEW software. The interface of the software application is provided with a field inside which, from the keyboard, the reference value of the angular displacement of the motor axis is entered. The virtual dial indicates the value of the angle traveled by the motor shaft, up to the desired angle (Figure 8).

The LabVIEW application communicates with the Arduino Uno platform through the serial port. The microcontroller on the development board generates a series of PWM pulses until the pointer needle reaches the desired angle. The Hall encoder, located on the shaft of the DC motor, monitors its movement. When the reference angle is reached, a signal is sent to the development board, which, in turn, sends a response to the application, having the effect of blocking the pointer needle at the reference angle in the application.

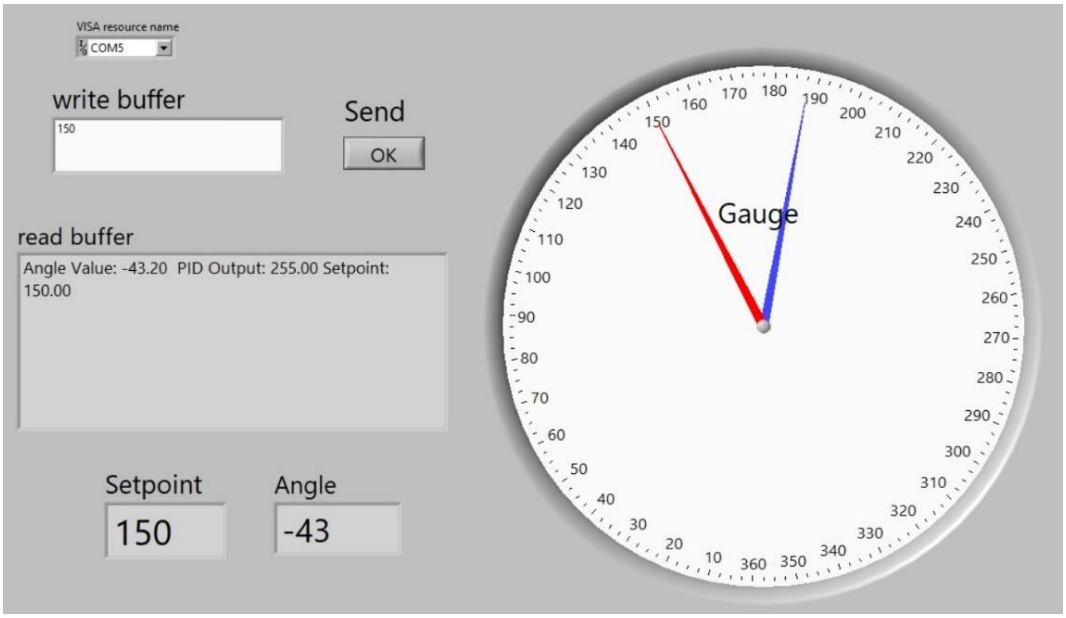


***Figure 8***. *LabVIEW operator interface in configuration mode*

The values to which we want the DC motor to move are typed in the "write buffer" field. The reference value to be reached is displayed in the "Set point" field. In the "Read buffer" field we will have "Angle Value" displayed, a parameter that represents the value of the angle where the indicator needle is located on the experiment stand, and "Set point" that represents the completed angle and to which the indicator needle must reach. The encoder sends the position to the Arduino board, and from the Arduino the signal is acquired and displayed in the interface.

The application code in LabVIEW is written using the "G" language specific to LabVIEW and which is completely visual, because it uses functional blocks instead of structured text (Figure 9).

## EXPERIMENTAL RESULTS

The tests performed were numerous and were carried out for different tuning parameters of the regulator. Table 1 shows the experimental values obtained for the parameters shows in the relation (2).

The values in the table were typed into the "write buffer" field located in the interface created in the LabVIEW professional development environment. The value reached by the indicator needle of the display device in the experimental stand is identical to that in the LabVIEW application.

It is important to mention that the supply of the Arduino Uno development board from which the supply of the L298N H-bridge and the motor-encoder assembly was made with a 9V and 3A power supply.

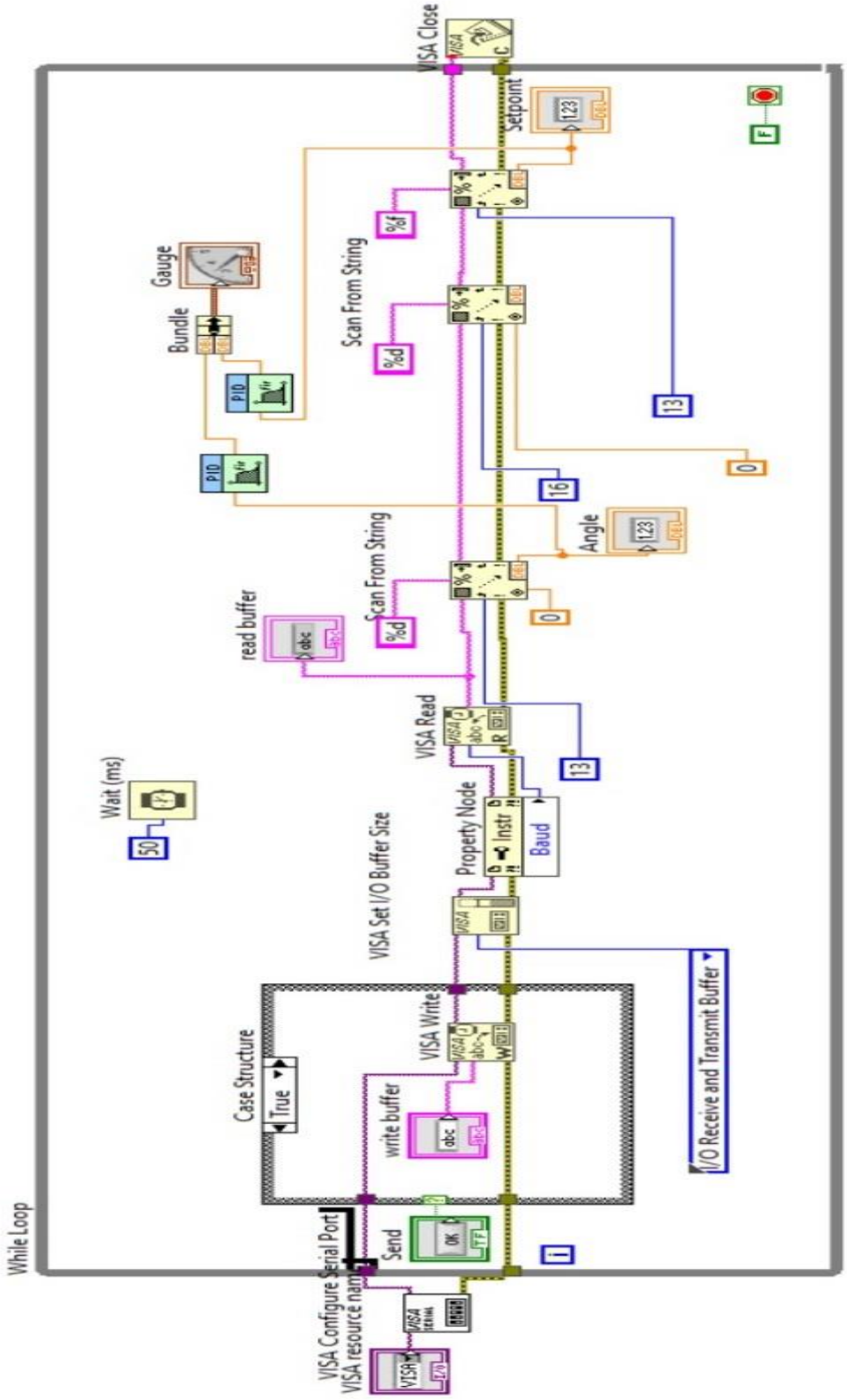


***Figure 9***. *Source code in LabVIEW*

The meaning of the blocks in this scheme is presented in the figures 10-16. The "VISA resource name" block is used in order to initiate the communication port. VISA configure serial port - a function block where the parameters of the serial port such as the baud rate (default 9600), data bits (default 8 bits), parity, and timeout can be configured. In order to increase the system's stability, it is necessary to set a small delay value.


***Figure 10***. *Virtual block utilized to enable the communication port*

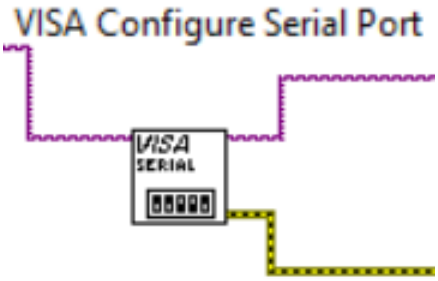


***Figure 11***. *Function block used for the communication port configuration (data transmission speed)*

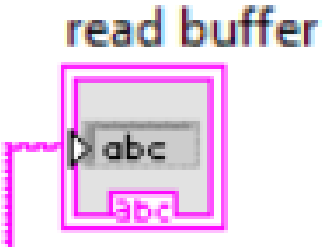


***Figure 12***. *Function block used for displaying the data from inside the buffer*

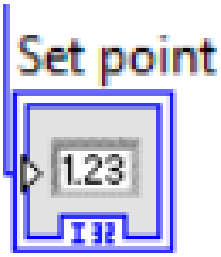


***Figure 13***. *Function block for setpoint value input*

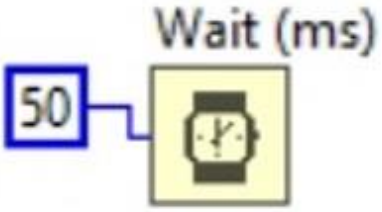


***Figure 14***. *Function block for the delay value setting*

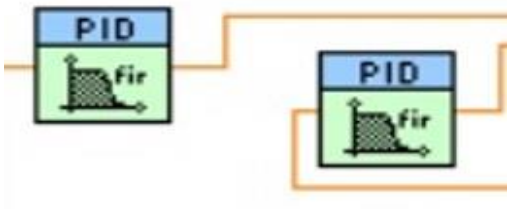


***Figure 15.*** *The LabVIEW PID function block containing the PID calculation algorithm*


***Figure 16**. Function block for displaying the real-time displacement of the indicating needle in gauge-style*

The tuning of the controller was done using the Ziegler-Nicols method. A multitude of experiments were performed for various values of the tuning parameters, some of them is presented in table 1.

***Table 1**. Experimental results for various values of the tuning parameters*

| **Ref. values [degree]** | **Value displayed [degree]** Kp=0,11[-] Ki=0,03[1/s] Kd=0,01[s] | **Value displayed [degree]** Kp=0,11[-] Ki=0,13[1/s] Kd=0,01[s] | **Value displayed [degree]** Kp=0,11[-] Ki=0,17[1/s] Kd=0,01[s] | **Value displayed [degree]** Kp=0,17[-] Ki=0,08[1/s] Kd=0,05[s] | **Value displayed [degree]** Kp=0,06[-] Ki=0,08[1/s] Kd=0,05[s] | **Value displayed [degree]** Kp=0,23[-] Ki=0,08[1/s] Kd=0,05[s] |
|---|---|---|---|---|---|---|
| 180 | 163,190 | 179 | 192-183 | 165,184 | 184 | 129,179 |
| 6 | 58,64 (unstable) | 57,63,60 | 70-47 | 60 | 51,63,60 | 77,55 |
| 200 | 146,198 | 199 | 200 | 185,207 | 210,199 | 152,202 |
| 30 | 28 | 29-31 (unstable) | 18-40 (unstable) | 48,21 | 31 (unstable) | 100,30 |
| 160 | 160 | 169 | 158-170 | 161 | 159 | 112,152,165 |
| 45 | 46 | 45 | 50-40 (unstable) | 48 | 44 | 82, 46 |
| 250 | 198,250 | 260 | 254 | 242,249 | 260 | 199, 253 |
| 77 | 75 | 76 | 62-90 (unstable) | 77 | 68-80 | 135,77 |
| 300 | 251,301 | 309 | 302-298 (unstable) | 277,304 | 297-304 | 240,300 |
| 90 | 99,84,89 | 84-101,89-92 | 105-85 | 101,83 | 100-80 | 135,88 |

After multiple attempts, the values of the PID tuning parameters were finally set to the following values, for which measurement errors were minimal:

$$K_p=0.11\ [-],\ K_i=0.08\ [1/s],\ K_d=0.05\ [s]. \quad (2)$$

The obtained final results are presented in table 2.

***Table 2***. *Experimental final results*

| **Reference value [degree]** | **Value displayed [degree]** | **Error = Value displayed - Reference value [degree]** |
|---|---|---|
| 180 | 181 | 1 |
| 60 | 59 | -1 |
| 200 | 204 | 4 |
| 30 | 25 | -5 |
| 160 | 160 | 0 |
| 45 | 43 | -2 |
| 250 | 252 | 2 |
| 77 | 76 | -1 |
| 300 | 302 | 2 |
| 90 | 86 | -4 |

## CONCLUSIONS

During the tests, the increase in system instability was observed at values towards 360 degrees of the reference signal. This leads to the idea of many more attempts to determine the optimal values of the PID parameters, which will be used in the source code.

Since the motor uses an incremental encoder to reach the desired value, it is important to reset the mount to 0 degrees after each use. Unlike absolute encoders, incremental encoders do not retain the value they reached during their last execution, and therefore the previously specified reset is required.

Based on the experimental results obtained, we can conclude that the system used, composed of Arduino Uno, H-bridge, motor and encoder, is suitable for educational use, for understanding the constructive, programming and assembly principles. These findings emphasize the importance of a careful and rigorous approach in selecting the right system for professional use.

## REFERENCES

[1] Ajaykumar, K.H., Akash, R.B., Koushik, S., Sachin, P., Arduino-PID Based PMDC Motor Angular Position Control System, Journal of Emerging Technologies and Innovative Research, volume 8, issue 8, pp 450-454, 2021

[2] Arifur, R., Syed, MA, Adaptive control of angular position & angular velocity for a DC Motor with full state measurable, International Journal of Engineering Research and Application (IJERA), vol.3, issue 4, 2013

[3] Autson, S., Kudelina, K., Vaimann, T., Rassolkin, A., Kallaste, A., Principles and methods of servomotor control: comparative analysis and application, Applied Science, 14, 2579, 2024

[4] Baranowski, J., Długosz, M., Mitkowski, W., Remarks about DC motor control. Archives of Control Sciences, 18(3):289–322, 2008

[5] Dlugosz, M., Mitkowski, W., Control System for tracking angular position DC servomotor, Międzynarodowa Konferencja z Podstaw Elektrotechniki i Teorii Obwodów IC-SPETO Conference, Poland, Wisła, 2010

[6] Georgescu, L., Electrical machines and actuators, UPG Ploiesti Publishing House, 2015 (in Romanian)

[7] Kaya, I., IMC based automatic tuning method for PID controllers in a Smith predictor configuration, Computers & chemical engineering, 28(3): p. 281-290, 2004

[8] Leonhard, W., Control of a Separately Excited DC Machine. Control of Electrical Drives, p. 77-96, 2001

[9] Matei, D., Negroiu, A., Design and implementation of an angular displacement control system with DC motor, project in the field of Robot Control System, year IV, AIA IFR, coordinator assoc.prof. G. Bucur, UPG Ploiesti, 2023

[10] Maung, M., Maung, M.L., New, C.M., DC Motor Angular Position Control using PID Controller with Friction Compensation, International Journal of Scientific and Research, 2018

[11] Meshram, P.M., Tunning of PID Controller using Zieglar-Nichols Method for Speed Control of DC Motor, IEEE International Conference on Advanced in Engineering, Science and Management, 2012

[12] Moise, A., Popescu, C., Robot driving systems. Basic structures (in Romanian), UPG Ploiesti Publishing House, 2015

[13] Naveenkumar, R., Krishna D.P., Low Cost Data Acquisition and Control using Arduino Prototyping Platform and LabVIEW. International Journal of Science and Research, 2(2): p. 366-369, 2013

[14] Petráš, I., Fractional-order feedback control of a DC motor. Journal of Electrical Engineering, 60(3): p. 117-128, 2009

[15] Popa C., New Approach in Modelling, Simulation and Hierarchical Control of Fluid Catalytic Cracking Process II - Hierarchical control, Revista de chimie, vol. 64, nr. 10, p. 1167-1171, 2013

[16] Rao, A.P.C., Obulesu Y., Babu C.S., Robust Internal Model Control Strategy based PID Controller for BLDCM. International Journal of Engineering Science and Technology, 2(11), p. 6801-6811, 2010

[17] Rubaai, A., Kotaru, R., Online identification and control of a DC motor using learning adaptation of neural networks. Industry Applications, IEEE Transactions on, 36(3): p. 935-942, 2000

[18] Saad, M., Amhebd, A., Al Sharqawi, M., Real Time DC Motor Position Control using PID Controller in LabView, Journal of Robotics and Control, volume 2, issue 5, sept. 2021

[19] Sailan, K., Kuhnert, K.D., DC Angular Position Control using PID Controller for the purpose of Controlling the Hydraulic Pump, International Conference on Control, Engineering & Information Technology, volume 1, pp 22-26, 2013

[20] Tian-Hua Liu; Yung-Chung Lee; Yih-Hua Crang, Adaptive controller design for a linear motor control system, Aerospace and Electronic Systems, IEEE Transactions, vol.40, no.2, pp.601-616, April 2004

[21] Wang, J., Luo, Z., Wang, Y., B. Yang, Assadian, F., Coordination Control of Differential Drive Assist Steering and Vehicle Stability Control for Four-Wheel-Independent-Drive EV, IEEE Trans. Veh. Technol., vol. 67, no. 12, pp. 11453–11467, Dec. 2018

[22] Zouari, F., Saad K.B., Benrejeb M., Adaptive Internal Model Control of a DC Motor Drive System Using Dynamic Neural Network. Journal of Software Engineering and Applications, 5(03), p. 168, 2012

[23] https://ro.huahaomotors.com/news/the-development-history-of-brushless-dc-motors-58213303.html

[24] https://www.sigmanortec.ro/Punte-H-Dubla-L298N-p125423236

[25] https://store.arduino.cc/products/arduino-uno-rev3

[26] https://bluetechs.wordpress.com/wp-content/uploads/2013/05/dc_motor_control_position.pdf

Received: July 2024; Revised: September 2024; Accepted: September 2024; Published: September 2024