Proceedings of CONF-FMCE 2026 Symposium: Artificial Intelligence and Smart Sensing for Mechanical and Electrical Systems
DOI: 10.54254/2755-2721/2026.BA36169

Design and Simulation of a Closed-Loop DC Motor Position Control System Based on Arduino

Chenxi Yang

School of Electronics and Computer Engineering, Chengxian College, Southeast University, Nanjing, China
chenxiyang0617@163.com

Abstract. The DC motor is affected by hysteresis characteristics, resulting in large steady-state errors and poor positioning accuracy with open-loop control; therefore, open-loop control is not suitable. To verify the PID closed-loop control logic in a cost-effective way, this study uses the Tinkercad simulation environment to build a test circuit that includes an Arduino main controller, L293D driver chip, DC motor, and adjustable potentiometer. The potentiometer is connected to the A0 analog pin of the microcontroller to simulate angle sampling, without mechanical coupling between the device shafts. First, the Arduino program reads the potentiometer value on pin A0, maps the 0–1023 range to 0–180 degrees, and calculates the error relative to the 90-degree target angle. PID output is restricted within the 0–255 PWM range and regulates motor rotation direction through L293D pins. The potentiometer serves as simulated angular feedback, and the setup excludes direct mechanical drive from the motor shaft to the potentiometer. Simulation test results show that the system can automatically adjust the PWM output magnitude and motor direction based on real-time errors, the closed-loop control logic operates stably, and it can follow the target angle to complete position adjustment. The significance of this study lies in the use of virtual simulation to debug the control scheme in advance, reducing the cost of processing physical components, quickly verifying the practicality of PID, shortening prototype debugging cycles, and saving experimental materials. On the other hand, the low-cost design approach based on Arduino can lower the barrier to implementing motor closed-loop control systems, providing a reference solution for position control system design in small electromechanical devices and teaching experimental equipment, and has practical significance for popularizing embedded motor closed-loop control in engineering practice.

Keywords: Arduino, DC motor, potentiometer-based feedback, PID control, Tinkercad

1. Introduction

A DC motor directly drives the armature linearly through electromagnetic force, without mechanical conversion. Its operation is affected by uncertain factors such as mechanical hysteresis, voltage fluctuations, and load variations. Open-loop control lacks feedback acquisition, so it cannot obtain the deviation between the actual and target angles in real time, nor can it autonomously correct the drive output. Consequently, it has shortcomings such as large positioning errors and poor anti-

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.BA36169

interference performance, making high-precision angular positioning difficult. In contrast, closed-loop control uses feedback devices to collect real-time actual position information, compares it with the set target to determine the deviation, and the controller dynamically adjusts output power based on the error to compensate for position shifts caused by external disturbances, thereby suppressing steady-state errors from the control principle standpoint.

As an open-source hardware platform, Arduino has the advantages of low development threshold and low hardware cost. Li Jianrong and others, working within the STEM education framework and using the Arduino platform, designed and produced experimental equipment that can collect data in real time and control reaction conditions via programs. This demonstrates that this main controller can significantly reduce the investment in device development, making it a preferred solution for the implementation of low-cost embedded control systems [1].

The PID algorithm is mature and reliable, with convenient parameter tuning, and is widely applied in the field of embedded closed-loop control. Yang Jiyu et al. applied neural network-optimized PID to UAV attitude regulation, demonstrating the practical value and stability of the PID algorithm in embedded closed-loop position control [2]. In research related to EGR DC motor actuators, Zhang Kexun et al. verified that adopting closed-loop PID control can optimize motor position control performance, achieve rapid step responses, and effectively reduce steady-state errors [3].

Traditional motor control experiments rely on physical components, which involve complex wiring, susceptibility to component damage, long debugging cycles, and high material consumption. Tinkercad allows users to build circuits, compile programs, and run simulations without physical hardware. This enables rapid verification of circuit logic and control algorithms, reduces the risks of physical experiments, and speeds up prototype iteration and parameter tuning [4].

Based on the above background, the research objective of this paper is to design a closed-loop position control system based on Arduino and L293D. This experiment aims to construct a Tinkercad hardware simulation circuit for DC motor angular position control and to verify the Arduino PID logic, L293D drive logic, and the simulated feedback loop. The main research content of this paper includes:

(1) Use Arduino to read the analog angle feedback signal from a potentiometer.

(2) Use the direction pins and PWM enable pins of the L293D to drive the DC motor.

(3) Monitor the target angle, position, error, and PWM output through the serial monitor.

(4) Explain the differences between Tinkercad simulation and a real mechatronic closed-loop system.

2. System architecture and control principles

2.1. Hardware architecture

The Tinkercad circuit hardware architecture includes four main components: an Arduino controller, an L293D H-bridge motor driver, a potentiometer for analog angle feedback, and a DC motor. Arduino serves as the core system controller, responsible for signal acquisition, data processing, and control command output. The L293D contains a built-in H-bridge structure, which functions to amplify weak signals and switch motor rotation direction; the potentiometer is connected to the A0 port, converting voltage signals into corresponding angles, generating a deviation when compared with the 90° set value, with the motor acting as the actuator to complete the rotation action.

The system hardware relies on the collaboration of each component to build a closed-loop chain: the potentiometer delivers an analog feedback signal to the main controller through the A0 channel.

DOI: 10.54254/2755-2721/2026.BA36169

The Arduino converts the analog signal to digital and compares it with the 90° reference. The controller then sends direction and PWM speed signals to the L293D, which uses its H-bridge to boost power and drive the motor. In the simulation scheme, the potentiometer is independently arranged and not mechanically coupled with the motor shaft, serving only as an equivalent position sensor to complete the logical verification of the closed-loop hardware architecture.

2.2. Principles of PID control

The PID controller calculates the control force based on the error, the integral of the error, and the derivative of the error. The absolute control value is limited by the PWM range, and the sign of the error determines whether the motor rotates forward or backward. A ±2-degree dead zone causes the motor to stop when it is close to the target position. The proportional component responds instantly to the real-time error, reducing the angle deviation quickly and speeding up system response—but it cannot eliminate steady-state errors. The integral component accumulates the total process error and continuously outputs a compensating amount, thereby eliminating static positioning errors; unreasonable values can easily lead to output saturation and system oscillation. The derivative component captures the rate of change of the error and predicts its trend, adjusting the control output in advance, effectively suppressing overshoot and jitter, and optimizing system dynamic performance. The three components jointly compute the control signal and adjust the PWM output magnitude. Combined with the set positional dead zone, this drives the motor to smoothly approach and stabilize at the 90° target angle. Due to its simple structure and convenient tuning, PID has become the mainstream solution for motor closed-loop control and is widely applied in high-precision position and speed regulation of various motors [5].

$$e(t)=\theta_{ref}(t)-\theta(t) \tag{1}$$

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{2}$$

$$PWM = \min(255, \max(0, |u(t)|)) \tag{3}$$

Table 1. Tinkercad program parameters

Parameter	Value	Meaning
targetAngle	90.0	Reference angle
Kp	4.0	Proportional gain
Ki	0.02	Integral gain
Kd	1.2	Derivative gain
dead band	+/-2 degrees	Stop range near the target

Table 1 presents the key parameter configurations for PID closed-loop control within the Tinkercad simulation system. The target angle is set to 90.0°, with a proportional coefficient Kp = 4.0, integral coefficient Ki = 0.02, and derivative coefficient Kd = 1.2, while a ±2° control dead zone is also established. The above parameters were repeatedly simulated and tuned, balancing system response speed and steady-state accuracy. The dead zone parameter is used to suppress frequent motor start-stop oscillations near the target position.

DOI: 10.54254/2755-2721/2026.BA36169

2.3. PWM speed and direction control

PWM (Pulse Width Modulation) is the core technology for stepless and smooth speed regulation of small DC motors. By changing the duty cycle of the pulse signal within a single cycle, the effective average voltage across the motor armature can be adjusted. The higher the duty cycle, the greater the average output voltage, and the faster the motor runs, enabling continuous linear adjustment of motor speed. This approach is efficient, responsive, and accurate [4]. In this system, the Arduino D9 pin outputs a PWM signal within the 0–255 range, corresponding to a 0%–100% duty cycle. Combined with the PID computation results, the output value is dynamically adjusted to achieve adaptive control of motor speed.

Motor direction is controlled by an H-bridge circuit built into the L293D. The D7 and D8 pins output high and low voltage combinations to switch current flow, which sets the motor's operating state [6]. When the D7 and D8 levels are inconsistent, the internal current direction of the H-bridge changes, causing the motor to rotate forward or backward. When both pins are high or both are low, the motor enters a braking stop state, thus accomplishing precise direction and start-stop control. This logic is simple, the response is stable, and it represents a classic solution for bidirectional control of small motors.

To reduce steady-state oscillation, the system uses a ±2° angle dead zone. In a PID position closed-loop system, small errors near the target position can trigger frequent fine adjustments by the controller, causing minor back-and-forth oscillations of the motor and frequent start-stop actions, which affect positioning stability [7]. By setting a reasonable angle dead zone, invalid control outputs corresponding to minor errors can be blocked. When the angular deviation is less than 2°, the PWM drive signal is cut off, allowing the motor to remain stably stationary. This effectively suppresses oscillations around the set point while ensuring positioning accuracy, significantly enhancing the system's steady-state stability.

3. System implementation based on Tinkercad

3.1. Circuit design and hardware connection

This study relies on the Tinkercad Circuits simulation platform to build a DC motor angle closed-loop control system. The simulation environment and device information related to the project are shown in Table 2. The system uses an Arduino Uno as the main controller, an L293D chip as the motor driver, plus a potentiometer and a DC motor to form the closed-loop hardware.

Table 2. Tinkercad environment and materials

Item	Content
Project topic	Closed-loop angular position control of a DC motor
Benchmark parameters	J=0.01, b=0.1, K=0.01, R=1, L=0.5
Source materials	04_Code source files, 03_Screenshots images, and the existing integrated Word report
Simulation tool	Tinkercad Circuits
Controller	Arduino Uno

DOI: 10.54254/2755-2721/2026.BA36169

Table 2. (continued)

Driver	L293D / L293E motor driver
Program file	04_Code/arduino_pid_tinkercad.ino
Board file	04_Code/Ingenious Bruticus.brd
Result screenshots	Three images under 03_Screenshots/Tinkercad

The detailed pin wiring specifications are listed in Table 3: The system uses a potentiometer to output an analog voltage, which is collected via the Arduino A0 pin and then converted into angle information; the D9 pin outputs a PWM signal to control the motor speed, while the D7 and D8 pins output direction control signals, with motor forward and reverse control implemented through the L293D. The system follows a closed-loop sequence: acquisition, computation, drive, execution, and feedback — and is used to test the PID position control logic.

Table 3. Main pin connections

Arduino pin	Connection	Function
A0	Potentiometer wiper	Read simulated angular feedback
D9	L293D enable pin	PWM speed command
D7	L293D input 1	Motor direction command
D8	L293D input 2	Motor direction command
5V/GND	Potentiometer and driver common reference	Power and ground reference

3.2. Arduino program design

The system control program is written using the Arduino IDE, implementing three functional modules: feedback acquisition, PID computation, and PWM output. First, the feedback value from the potentiometer is periodically read through the A0 pin, and the ADC sampling data ranging from 0 to 1023 is mapped to an angular range of 0° to 180°. The program then calculates the error between the current and target angles, and applies the PID algorithm to generate a control value. This value is then limited and converted to a PWM signal. After the absolute value is clipped within upper and lower limits, it is restricted to the standard PWM range of 0–255. Finally, the speed-regulating PWM signal is output through the D9 pin, while the D7 and D8 pins output a combination of high and low levels to achieve forward and reverse motor control.

3.3. Simulation process and experimental procedures

This experiment uses Tinkercad to test closed-loop angle control of a DC motor. The specific procedure is as follows:

(1) Build a closed-loop control circuit consisting of an Arduino, L293D driver, potentiometer, and DC motor;

(2) Develop and upload the PID control program;

(3) Start the simulation, adjusting the potentiometer to simulate positional changes;

(4) Use the serial monitor to record the target angle, feedback angle, error value, and PWM output in real time;

DOI: 10.54254/2755-2721/2026.BA36169

(5) Adjust the PID parameters according to the system's response;
(6) Verify the system's stability and position control performance.

During the experiment, the target angle was set to 90°, and the serial communication baud rate was set to 9600 bps. By observing changes in the angle, the error convergence process, and PWM output variations, the system's dynamic response and steady-state control performance were evaluated. The results provide a basis for further analysis and validation.

4. Experimental results and analysis

4.1. Circuit verification results

The wiring diagram (Figure 1) shows the complete circuit of the Arduino, L293D, DC motor, and potentiometer. Figure 2 verifies the analog reading, angle mapping, PID calculation, anti-integral saturation limit, PWM output, and direction control. The serial monitor displays the target value, position value, error value, and PWM value. The greater the error, the stronger the PWM output; when the error is within ±2 degrees, the motor stops rotating (see Figure 3).

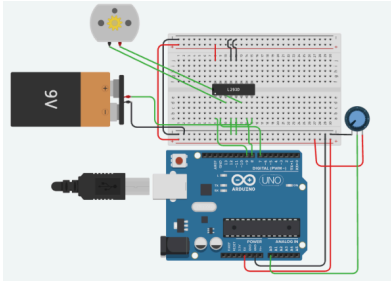

Figure 1. Tinkercad Circuit wiring

```
// DC Motor Position Control in Tinkercad
// Arduino + L293D + DC motor + potentiometer feedback
// The potentiometer is used as a simulated angle sensor.

const int potPin = A0;

const int enPin = 9;   // L293D pin1, PWM speed control
const int in1Pin = 7;  // L293D pin2
const int in2Pin = 8;  // L293D pin7

// Target angle
float targetAngle = 90.0;

// PID parameters
float Kp = 4.0;
float Ki = 0.02;
float Kd = 1.2;

float integral = 0.0;
float lastError = 0.0;

unsigned long lastTime = 0;

void setup() {
  pinMode(enPin, OUTPUT);
  pinMode(in1Pin, OUTPUT);
  pinMode(in2Pin, OUTPUT);

  Serial.begin(9600);

  lastTime = millis();

  Serial.println("DC Motor Position Control Started");
  Serial.println("Target,Position,Error,PWM");
}

void loop() {
  unsigned long now = millis();
  float dt = (now - lastTime) / 1000.0;
```

Figure 2. Tinkercad Arduino PID code

DOI: 10.54254/2755-2721/2026.BA36169

```
DC Motor Position Control Started
Target, Position, Error, PWM
90.00, 0.00, 90.00, 255
90.00, 0.00, 90.00, 255
90.00, 43.00, 47.00, 138
90.00, 100.00, -10.00, 106
90.00, 97.00, -7.00, 22
90.00, 151.00, -61.00, 255
90.00, 172.00, -82.00, 255
90.00, 122.00, -32.00, 70
90.00, 75.00, 15.00, 114
90.00, 43.00, 47.00, 225
90.00, 0.00, 90.00, 255
90.00, 0.00, 90.00, 255
90.00, 0.00, 90.00, 255
```

Figure 3. Tinkercad serial monitor result

Results Analysis: The Tinkercad experiment tested the full software chain—from sensor input, to controller computation, and on to motor driver output. A0 provides analog angle feedback, PWM sets the control level, and the direction pin controls forward/reverse rotation.

In this project, potentiometers are used as analog feedback. To implement a true hardware solution, encoders, mechanically coupled potentiometers, or other angle sensors are needed to form a closed loop around the actual motor shaft.

4.2. PID response characteristics

The optimal PID parameters determined in this experiment are Kp = 4.0, Ki = 0.02, and Kd = 1.2. The target positioning angle is 90°, with a ±2° shutdown dead zone set. The system responds quickly and settles steadily at the target.

The proportional component responds to large initial errors by outputting a full-scale PWM signal, driving the motor at full speed toward the target. This cuts the adjustment time and speeds up error correction. The integral component continuously accumulates the total error throughout the process, effectively eliminating static positioning residual errors, preventing fixed-position drift, and enhancing steady-state positioning accuracy. Meanwhile, the integral limit strategy effectively avoids integral saturation under large deviation conditions, preventing control output from exceeding limits. The derivative component tracks the error's rate of change, predicts its trend, and adjusts the output ahead of time. This reduces overshoot and oscillations near the target, and improves dynamic stability [7].

The setting of a ±2° angular dead zone is key to suppressing steady-state oscillations. It can effectively shield minor invalid adjustments near the target position, preventing frequent motor start-stop cycles and small-amplitude reciprocating movements, thereby significantly enhancing steady-state positioning stability. Serial timing data indicate that the system can accurately identify positive and negative errors and automatically switch the motor's forward and reverse states. The ultimately measured angle stabilizes within the 90°±2° range, with no significant overshoot or sustained oscillation. The PID parameters are reasonably tuned, and both the dynamic response speed and steady-state control accuracy meet the design specifications.

DOI: 10.54254/2755-2721/2026.BA36169

4.3. Limitations of the simulation model

Idealized Feedback Structure: The simulation feedback method is somewhat idealized, with the potentiometer and motor shaft separated and values adjusted manually; the physical system requires coaxial installation of the potentiometer or pairing it with an encoder to capture the actual rotation angle, resulting in structural differences between the simulation feedback form and the physical hardware.

Motor simplification: the Tinkercad motor model ignores friction, inductance changes, and load disturbances. In reality, friction alone can hurt positioning accuracy—though pre-compensation can help. The simulation also lacks mechanical hysteresis, so parameters must be re-tuned for a physical prototype [8].

Idealized Driving Devices: The L293D simulation model does not account for inherent device losses such as voltage drop and conduction resistance, while the actual chip has power consumption and voltage drop. Under the same PWM duty cycle, the speed of a physical motor is lower than the simulation result.

Missing Environmental Factors: The simulation environment assumes constant temperature and supply voltage, while in practice, power ripple and environmental temperature drift continuously exist. Temperature changes can alter device electrical characteristics, causing systematic measurement deviations. As pointed out by Dong Zipeng et al. in meteorological observation studies, environmental temperature fluctuations can lead to a systematic bias in observations, such as 'underestimation at high temperatures and overestimation at low temperatures,' interfering with the inherent correlations in the data. Therefore, PID parameters obtained from calibration under ideal constant-temperature and stabilized-voltage conditions cannot be directly applied to a physical control system [9].

5. Conclusion

This study used Tinkercad to build a DC motor position closed-loop control system with an Arduino controller, an L293D driver, and a potentiometer for simulated angle feedback. The design and simulation tests were completed on this platform. The paper first analyzes the various issues inherent in direct open-loop control of DC motors and identifies core control methodologies such as PID closed-loop regulation, PWM stepless speed control, drive chip level commutation, and reducing jitter within the angle dead zone. The simulation platform combines hardware and software to achieve precise 90° angle positioning. The system reads angle data converted from the potentiometer via the A0 interface of the microcontroller and calculates the control quantity in real-time through PID computation; the D7 and D8 pins are used to switch the motor's forward and reverse rotation based on high and low levels, while the D9 pin outputs signals with different duty cycles to adjust motor speed. An additional ±2-degree stopping range is set to prevent the motor from oscillating around the target point, resulting in a coherent and stable closed-loop operation. Simulation results show that the system compares the current angle with the target, adjusts motor output and direction accordingly, reduces position error quickly, and holds the target stably. This fixes the large errors and instability seen in open-loop control.

There are still several imperfections in this simulation experiment. Components and motors in the simulation are idealized models that do not account for real-world issues such as motor friction, chip power loss, voltage fluctuations, or environmental interference, meaning the control parameters tuned in the simulation cannot be directly applied to physical devices. In the experiment, the potentiometer and motor shaft are mounted separately and only manually adjusted to simulate

DOI: 10.54254/2755-2721/2026.BA36169

angles, which differs significantly from real device structures such as encoders or coaxial potentiometers used for actual position detection. The control program utilizes only a basic PID with fixed parameters and cannot automatically adjust according to external conditions, causing reduced control performance under load variations or external disturbances. Furthermore, the study is limited to virtual simulation without constructing a physical prototype for real-world testing, lacking operational data from actual hardware for comparison.

To address these shortcomings, further research can be optimized in several ways. First, set up a physical hardware test bench and replace the potentiometer with an optical encoder to capture the motor's actual rotation angles, and recalibrate PID parameters considering real-world losses and friction, comparing simulation and physical data to analyze performance disparities. Second, optimize control algorithms by adopting incremental PID, fuzzy PID, or other improved control strategies, combined with friction compensation logic, to enhance system adaptability and anti-interference capability under varying loads and non-linear conditions. Finally, run comparative tests with different supply voltages or added loads to gather data across a range of operating conditions. This low-cost approach could be extended in several ways: applied to small actuators or teaching kits, combined with machine learning for auto-tuning, or scaled up to multi-motor synchronization. Utilizing simulation to pre-validate algorithms can save materials and shorten debugging cycles, making it highly suitable for motor control research and education.

References

[1] Li, J. R., Chen, C. H., & Deng, F. (2026). Design and teaching practice of spectrophotometer based on open-source platform under STEM vision: A case of digital exploration on chemical equilibrium shift. *Journal of Education and Equipment Research*(6), 8–14.

[2] Yang, J. Y., & Su, L. (2026). UAV control system based on embedded structure and neural network PID. *Machine Design and Manufacturing Engineering, 55*(4), 41–45.

[3] Zhang, K. X., Chu, Q. H., Zhang, C., Jiang, C. F., Hao, S. G., & Li, J. (2013). Research on dual closed-loop position control algorithm for DC motor actuator of EGR system. *Chinese Internal Combustion Engine Engineering*, TK403, 45–51.

[4] Liang, J. H., & Zhou, K. (2022). Research on experimental teaching of motor PID control based on Tinkercad virtual simulation. *Laboratory Science, 25*(4), 89–92.

[5] Li, M. X. (2025). Research on control algorithm of hoist induction motor based on PSA. *Mining Machinery, 53*(1), 26–29.

[6] Huang, C., & Chen, J. (2023). Analysis on forward and reverse control logic and experiment of motor based on L293D H-bridge driver. *Mechanical & Electrical Engineering Technology, 52*(9), 139-142.

[7] Wang, H., & Li, Z. (2022). Research on oscillation suppression of DC motor PID position control based on dead zone correction. *Techniques of Automation and Applications, 41*(11), 23-27.

[8] Shao, X. J., Kang, P. F., Zhang, J. G., et al. (2024). Research on new control strategy of DC motor position servo system. *Control Engineering of China, 31*(9), 1537-1543.

[9] Dong, Z. P., Li, X. M., Peng, Y., et al. (2025). Impacts of radiosonde drift on temperature differences between microwave radiometer and radiosonde observations. *Meteorological Monthly, 51*(5), 529-541.