Motion Control Lab 2024: A barrier gate with a DC motor

1. Overview

The barrier gate is quite common in the car park. Generally, it is composed of a motor, a rotating bar, and a controller. In this lab, you are requested to design a mini barrier gate that can rotate 180 degrees instead of 90 degrees shown in Figure 1, and make it run smoothly with any given motion profile.

Figure 1. A barrier gate

Feel a bit hard? Don't be panic, you need not design it from the beginning and we have chosen some hardware with very specific tutorials for you.

2. Preliminaries

This experiment is split into two sections: **a physical experiment** and **a simulation experiment**. To complete this experiment, the following items need to be prepared in advance.

2.1 Simulation Experiment

To complete the simulation experiment, you need to prepare the following:

- MATLAB, which can be downloaded from https://software.sjtu.edu.cn/List/MATLAB/R2023a
- Function packages: Simulink, Simscape, and Simscape Electrical.

2.2 Physical Experiment

A box of hardware will be given to you with the listed components:

Hardware	Usage
ESP-32	Controller
DC motor driver with L298N chip	Driving the DC motor, changing the current and direction
12V DC Motor with Gear Reduction and Hall encoder	The Hall encoder generates a speed feedback signal and gears are to decelerate the motor.
12V DC power supply	Generating a DC power.

Table 1. Hardware description

3. Tutorials

This part will **NOT** tell you how to do it step by step but only some necessary basic knowledge. For more detailed instructions, we are confirmed you can get them by reading the friendly manual and searching the friendly web.

3.1 DC Motor

Specification	Value
Rated Voltage	12 V
Rated Speed	9 RPM
Rated Current	≤1 A
No-load Speed	12 RPM
No-load Current	≤120 MA
Gear ratio	810 :1
Hall encoder output	11 waves per revolution
Rated Torque	35 Kg.cm

Table 2. Motor specification

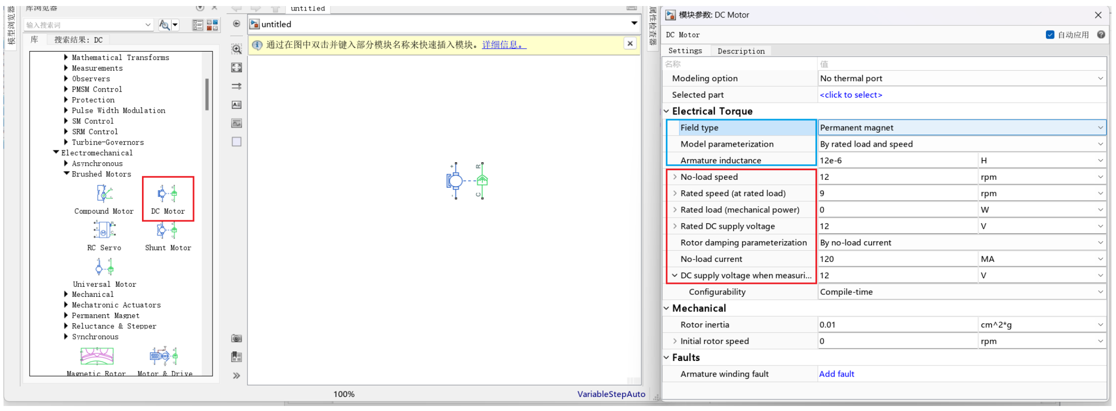
This motor is a permanent-magnet motor, which means the magnetic field is not generated by the current. Recall what you have learned about the DC motor. The armature voltage is set to change the speed and the current direction is set to reverse the motor.

You can count the hall encoder output waves to get the speed of the motor and complete closed-loop control.

For more information about the motor, please visit https://item.taobao.com/item.htm?id=559632206123

In the simulation experiment, the above motor can be simulated by the DC motor shown in the left part of Figure 2.

The parameters of the simulated motor in the blue box should be set according to Figure 2, whereas, the parameters in the red box must be consistent with the provided motor. (**The parameter settings in the red box of Figure 2 may not be correct.**)



模块参数: DC Motor

DC Motor

Settings | Description

名称	值	
Modeling option	No thermal port	
Selected part	<click to select>	
Electrical Torque		
Field type	Permanent magnet	
Model parameterization	By rated load and speed	
Armature inductance	12e-6	H
No-load speed	12	rpm
Rated speed (at rated load)	9	rpm
Rated load (mechanical power)	0	W
Rated DC supply voltage	12	V
Rotor damping parameterization	By no-load current	
No-load current	120	MA
DC supply voltage when measuri...	12	V
Configurability	Compile-time	
Mechanical		
Rotor inertia	0.01	cm^2*g
Initial rotor speed	0	rpm
Faults		
Armature winding fault	Add fault	

Figure 2. Information about the simulated DC motor

3.2 DC Motor Driver

Firstly, a frequently asked question is why a driver is needed and the motor cannot be controlled by ESP32 itself. The answer is quite simple, the ESP32 is just unable to provide enough power to drive the motor (almost all single-chip microcomputers are unable to do that, either). The maximum DC Current per I/O Pin is 20 mA, meaning it could only be able to act as a logical output. The DC driver is to isolate the logic from the power part.

The speed control is simple. The ESP32 has a PWM output and you can change the duty cycle to adjust the armature voltage.

The direction control is a bit harder and accomplished by an H-bridge circuit. As Figure 3 shows, the motor reverses when switching S1, S4, and S2, S3. For more about the H-bridge circuit, please refer to https://en.wikipedia.org/wiki/H-bridge.

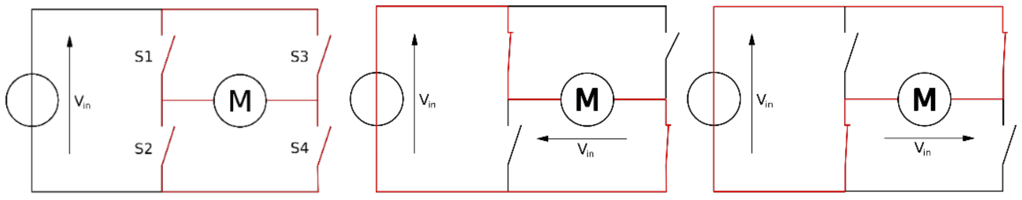


Figure 3. H-bridge circuit

The pin description of the L298N chip is listed in Figure 4. Additionally, the end of the power cord with a gray line should be connected to the “12V Power” pin of the L298N chip, and ESP-32 can be connected to the “5V Power” pin to obtain power. For more information about the L298N chip, please visit https://detail.tmall.com/item.htm?id=598155578199

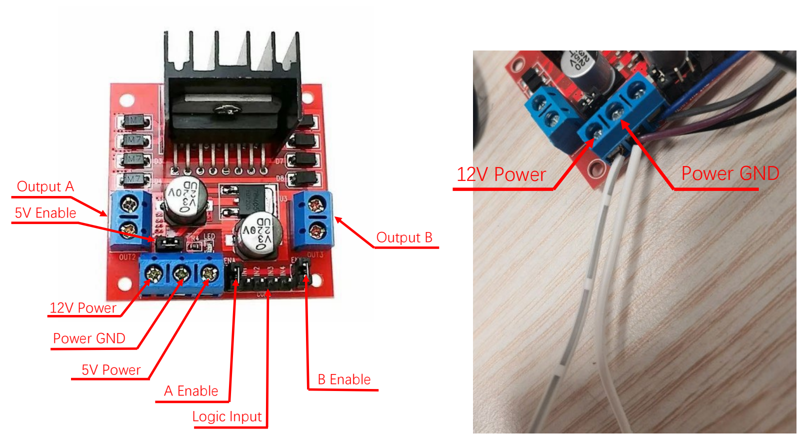


Figure 4. Pin description of the L298N chip

3.3 ESP-32 and Arduino IDE

ESP32 is a system on a chip that integrates the following features:

- Wi-Fi (2.4 GHz band)
- Bluetooth
- Dual high performance Xtensa® 32-bit LX6 CPU cores
- Ultra Low Power co-processor
- Multiple peripherals

Powered by 40 nm technology, ESP32 provides a robust, highly integrated platform, which helps meet the continuous demands for efficient power usage, compact design, security, high performance, and reliability.

You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

In this lab, it is highly recommended to program with Arduino IDE since you can easily find tutorials with high quality (E.G. https://www.bilibili.com/video/BV1YW411Z76E). A USB driver is necessary before starting programming. Configuring the IDE with "ESP32 Dev Module" (instructions for installing ESP32 board in your Arduino IDE) and your port, and you can start your lab.

Since hardware interrupt will be needed to get the speed of the motor, recall what you have learned in *EE222 Embedded System and Interface*. For demo codes, please refer to https://zhuanlan.zhihu.com/p/39452123.

The Arduino compiles codes with AVR-GCC 7.3.0, meaning you can develop it with C++ language. However, its compiling behavior may be a little different:

1. The setup() function is to initialize your program and the loop() function loops consecutively when running.
2. The communication between your PC and ESP-32 is based on TTL Serial Port, meaning you need to use Serial.print() rather than printf() for debugging, and do not forget to write Serial.begin() in your setup() function.
3. Arduino built-in functions could be found at: Arduino Reference - Arduino Reference. Data types, for example, string, are not the same with STL C++. If the STL library is

needed, see reference at https://www.arduino.cc/reference/en/libraries/arduinostl/.

A frequency counter library for ESP-32 is provided, click here. It counts the numbers of pulses on a specified pin during a fixed time frame using native interrupts and timers.

3.4 OLED Display Screen

An OLED display screen (0.96 inch) is provided. It has four pins and can be connected to ESP-32 by I2C (https://zhuanlan.zhihu.com/p/566224455). You can display the velocity curves obtained by trapezoidal and S-curve velocity profiles on the provided screen through programming.

4. Tasks and Requirements

Tasks:

1. Correctly connect the circuit.
2. Design a closed-loop control algorithm, and utilize Pulse Width Modulation to control the motor for generating Trapezoidal and S-curve velocity profiles.
3. Display the velocity profiles in real-time on the OLED screen.
4. Bluetooth or WiFi must be used, and the specific functionalities can be customized.
5. Simulate using Simulink according to the actual circuit connected, including the calculation of the parameter Rated load.
6. Adjust the parameters of your controller to make the motor run smoothly with Trapezoidal and S-curve velocity profiles.
6. Analyze the effects of Proportional, Integral, and Derivative terms by PID controller tuning.
7. Identify the transfer function of your control system.

Presentation:

- **Within 3 minutes.**
- **Only the physical experiment** needs to be demonstrated in this presentation.
- Focus primarily on implementation methods and effects.
- Demonstrating your experimental results live is better than playing a video.

Documents you need to submit:

A PDF lab report is required with:

- Your design methods for simulation experiments, step by step.
- Trapezoidal and S-curve velocity profiles.
- The analysis and effects of P/I/D.
- The concrete identification process of the transfer function.
- Problems you occurred, and how did you solve them.
- Written in English, and GPT is forbidden.

Your Arduino codes and Simulink slx model.

5. Grading Criteria

Presentation (50%)			
Physical experiment	Hardware connection	30	80
	Trapezoidal and S-curve velocity profiles	20	
	OLED Screen speed map	20	
	Bluetooth or Wi-Fi connection[1]	10	
PPT	Presentation effect[2]	20	20
Report (50%)			
Simulation experiment	Simulation model	10	90
	Close loop control algorithm	20	
	Trapezoidal and S-curve velocity profiles	20	
	Analysis of P, I, and D	20	
	Identification of the transfer function	20	
Report writing	Detail and accuracy	10	10

1: If both Bluetooth and Wi-Fi are implemented, you will receive a bonus of 10 points.

2: It is recommended to demonstrate live what you achieved instead of playing a recorded video.