**MEC 104 Report**

**Lab 2 Smart Car**

Group Number: A32

Group Member:

|  |
| --- |
| Zhongnan Li 1931254 |
| Chenruiyuan Yu 1928118 |
| Ding Wang 1927804 |

***Abstract***

*The purpose of this experiment is to design a smart car. This smart car uses Arduino Uno, Driver: L239D and two DC motors to achieve five functions: fast forward, slow forward, fast backward, slow backward and stop, and can pass Use IR remote controller to control and detect the safety distance to decelerate or stop, and an LED warning light will also light up accordingly. We use Thinkercad this software to design the smart car circuit and add code to realize the test of the simulated smart car. In the end, we realized the design of the smart car and learned more about regulating multi-variable control.*

**Content List**

**[1 Introduction 3](#_Toc2106244877_WPSOffice_Level1)**

**[2 Main Part 3](#_Toc2106244877_WPSOffice_Level1)**

[2.1 Theory](#_Toc2106244877_WPSOffice_Level1) [3](#_Toc2106244877_WPSOffice_Level1)

[2.2 Circuit Design](#_Toc508901442_WPSOffice_Level1) [6](#_Toc508901442_WPSOffice_Level1)

[2.3 Experimental Method](#_Toc1826653340_WPSOffice_Level1) [7](#_Toc1826653340_WPSOffice_Level1)

[2.4 Results](#_Toc136467868_WPSOffice_Level1) [8](#_Toc136467868_WPSOffice_Level1)

[2.5 Discussion](#_Toc102922480_WPSOffice_Level1) [10](#_Toc102922480_WPSOffice_Level1)

[1) Remote control](#_Toc508901442_WPSOffice_Level2) [11](#_Toc508901442_WPSOffice_Level2)

[2) Ultrasonic sensor](#_Toc1826653340_WPSOffice_Level2) [11](#_Toc1826653340_WPSOffice_Level2)

[2.6 Further study](#_Toc1093785525_WPSOffice_Level1) [11](#_Toc1093785525_WPSOffice_Level1)

**[3. Conclusion](#_Toc136467868_WPSOffice_Level2)** **[12](#_Toc136467868_WPSOffice_Level2)**

**[4. Author list and contributions](#_Toc102922480_WPSOffice_Level2)** **[12](#_Toc102922480_WPSOffice_Level2)**

**[5. Reference](#_Toc1093785525_WPSOffice_Level2)** **[12](#_Toc1093785525_WPSOffice_Level2)**

**[6. Appendix](#_Toc793300355_WPSOffice_Level2)** **[12](#_Toc793300355_WPSOffice_Level2)**

1. **Introduction**

In this experiment, we tried to control the operation of multiple elements through logic. It is realized by adding ultrasonic distance measurement and IR detection to the ARDUINO which controls motors through a program. In this process, we became familiar with the ARDUINO-based control method and the details of multiple element control. the factors that need to be considered for the operation of a system under complex scenarios are constantly increasing. The once simple single control method can no longer be applied to complex situations. Take the display as an example. In the past, a single button or IR could complete the basic requirements, but if you want to add more personalized operations such as unmanned automatic turn off, more factors need to be tested and considered.

Through distance detection and IR, we have achieved the control of the movement of the trolley. Although it is not an ideal perfect real-time control, it can be achieved for control with relatively small accuracy requirements. In theory, the error of control time is about 0.1s and the error of distance is about 1cm per meter. The car can perform different speeds of forward and backward and brake; in the case of forward, it will slow down or brake when it is too close. In addition, we also tried to add button controls to make it run in more complex situations, but it worked imperfectly into this system. This part of the content will be explained in further study.

1. **Main parts**
   1. **Theory**

In this part, theoretical background in terms of some basic components of the circuit will be introduced first, and then we will explain several functions used in the experiment. Finally, some of expected results will be anticipated.

**H-bridge/ L239D Driver**: To make the DC motor rotate in both forward and backward direction, a device called H-bridge should be used in this experiment. The working principle of H-bridge will be introduced below. The H-bridge motor drive circuit includes 4 transistors and a motor. To make the motor run, a pair of triodes on the diagonal must be turned on. According to the conduction conditions of different transistor pairs, current may flow through the motor from left to right or from right to left, thereby controlling the steering of the motor.

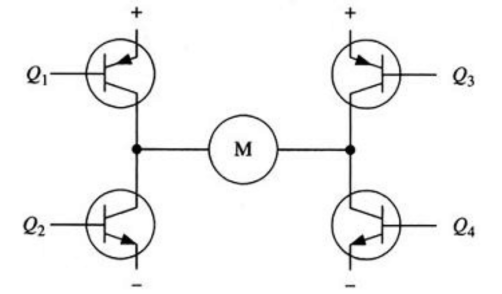


Figure 1: H-bridge motor drive circuit

For example, when the transistors Q2 and Q3 are turned on, the current flows from the positive pole of the power supply through Q3 through the motor from right to left to drive the motor to rotate in the forward direction, thereby controlling the smart car to move forward; when the transistors Q1 and Q4 are turned on, the current flows from the power supply The positive pole flows through Q1 from left to right through the motor to drive the motor to reverse rotation, thereby controlling the smart car to retreat.

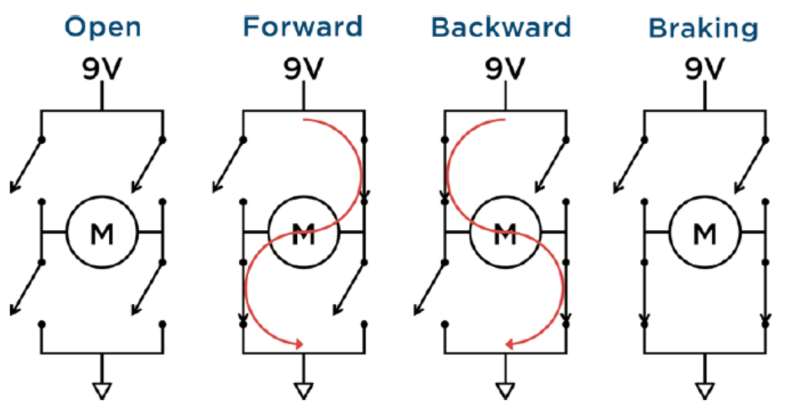


Figure 2: H-bridge inner Structure

As the equipment for this experiment, the L293D H-bridge driver has 16 pins, and their functions are as follows.

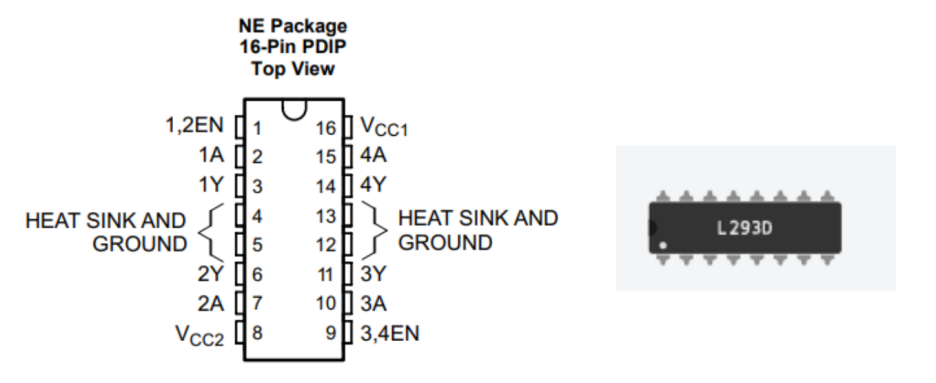


Figure 3: Pin function of H-bridge

GND (Pins 4, 5, 12, 13): These four pins in the middle place should connect to ground between 5V and 9V supply.

VCC2 (Pin8): VCC2 supplies the motor current by connecting to 9V.

VCC1 (Pin 16): VCC1 supplies the chip’s logic by connecting to 5V.

1Y and 2Y (Pins 3 and 6): These two pins are the outputs from the left driver, which should connect to motor wires.

1A and 2A (Pins 2 and 7): These two pins control the states of the switches on the left, which should connected to I/O pins on the Arduino.

1, 2EN (Pin 1): This pin enables or disables the left driver, which should connect to a PWM pin on the Arduino.

3Y and 4Y (Pins 11 and 14): These two pins are the outputs from the right driver, which should connect to another motor wires.

3A and 4A (Pins 10 and 15): These two pins control the states of the switches on the right, which should connected to I/O pins on the Arduino.

3, 4EN (Pins 9): This pin enables or disables the right driver, which should connect to a PWM pin on the Arduino.

In this experiment, we use H-bridge to control the directions of two DC motors by codes controlled by Arduino. More details will be explained in circuit design part.

**Romote control**: In the experiment, we need to achieve romote control by using an IR remote controller. A typical infrared communication system requires an IR transmitter and an IR receiver. All IR receivers have three pins: signal, ground, and Vcc as shown below.

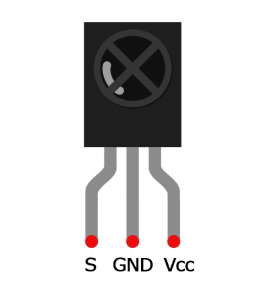


Figure 4: IR receiver

In infrared signal modulation, the encoder on the infrared remote control converts the binary signal into a modulated electrical signal. The electrical signal is sent to the sending LED. The emitting LED converts the modulated electrical signal into a modulated IR light signal. Then, the infrared receiver demodulates the infrared light signal and converts it back to binary, and then transmits the information to the microcontroller.

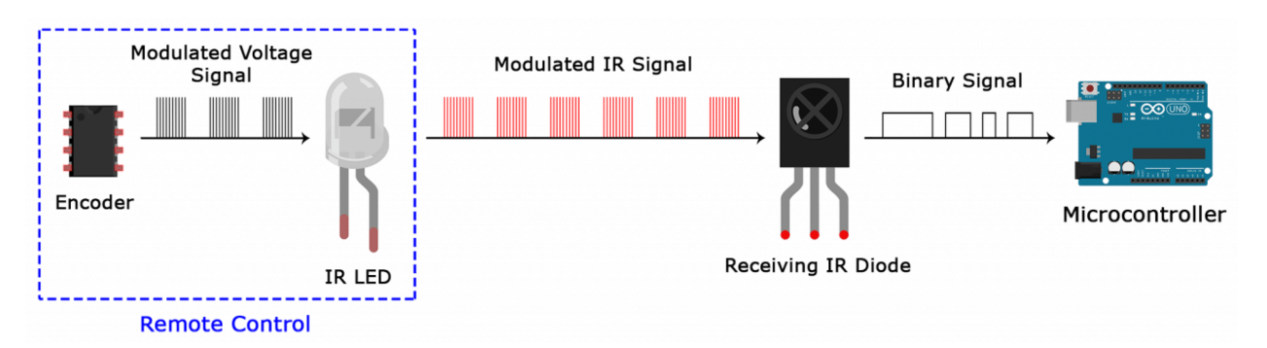


Figure 5: Process of the infrared signal modulation

Each time we press a button on the remote control, a unique hexadecimal code is generated. This is the information that is modulated and sent to the receiver via IR. To decrypt which key was pressed, the receiving microcontroller needs to know the code corresponding to each key on the remote control.

Therefore, we use the hexadecimal number displayed on the serial display to determine which key is pressed, to further control the smart car.

**Ultrasonic sensor**: The principle is that the ultrasonic sensor emits a certain frequency of ultrasonic waves, propagated by the air medium, and reflects back after reaching the measurement target or obstacle. After the reflection, the ultrasonic receiver receives the pulse. The time it takes is the round-trip time. The round-trip time is related to the propagation of ultrasonic waves. The distance of the journey is related. Since the ultrasonic speed is known, the distance from the obstacle is calculated by time.

**Expected results**: We expected to see two DC motors to spin in two directions according to our wishes. When we issue a command to make the smart car move forward, the Arduino controls the motor to rotate forward through the H-bridge to realize the forward function. When the command is issued to make the smart car move back, the Arduino controls the motor to rotate in the reverse direction through the H-bridge to realize the backward function. And can set the motor rotation speed, so that the speed of forward and backward changes. Of course, the function of stopping the smart car can also be realized with the help of the H bridge.

* 1. **Circuit Design**

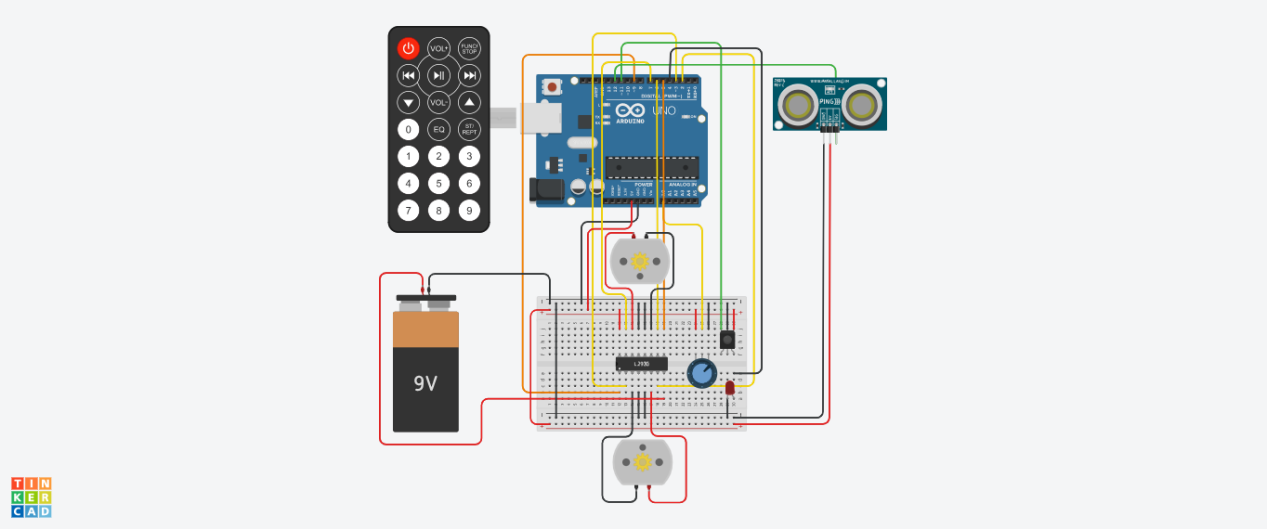
****

Figure 6: experiment circuit

The circuit above is what we built to achieve the smart car function. The specific circuit connections will be listed below. First, as the output terminals of Arduino, PWM pins 5 and 9 are respectively connected to pins 1 and 9 of the H-bridge to control the switching of the motor. The pins 2, 3, 6, and 7 on the Arduino are used as output terminals to connect the pins 2, 7, 10, and 15 of the H-bridge to control the rotation direction of the two motors. Pin 4 on the Arduino is connected to the LED as a distance warning light; pin 11 is connected to the IR transmitter; and pin 12 is connected to the ultrasonic rangefinder. The analog port A0 is connected to a potentiometer, and the 5V voltage output terminal and ground terminal are connected to the corresponding positions on the breadboard. Secondly, the connection between the H bridge and the two motors is the same as mentioned in theorem part.

* 1. **Experimental Method**

**Code structure**

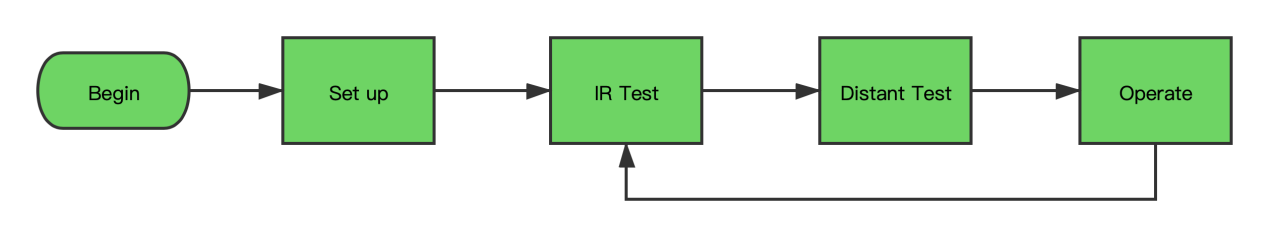
****

Figure 7: Flow chart about the whole process



Figure 8: Flow chart for operation

**Problems during debugging**

Generally, the loop used by ARDUINO and the logic used by the microcontroller are slightly different from the logic of other programming languages we are accustomed to. It takes a lot of time to get used to this programming method.

Although we do not fully understand and master the relevant knowledge, it is enough to support us in this task.

In detail, first, when considering the control priorities of different elements, the original assumption is equal.

But because it was too complicated and even affected the speed of operation, we finally chose to take IR as the main body, in which the operation was limited by the distance and gave up the extra button control.

Second, when distance control is added to a single IR control, the distance will be judged again only when the remote-control signal is changed.

This makes distance detection lose its meaning.

We add the judgment of the IR signal to the judgment of the change of the distance situation, so that the control ability of the distance detection can affect the operation.

This is an important step to achieve multi-element control.

Third, let the decelerating car light up the warning light. We tried to directly control the small light through a simple high level and low level, but in fact, this function cannot be achieved continuously.

In the end, we wrote the control of the light into the control of the motors. And differentiated the originally merged forward control into the deceleration function.

Fourthly, in view of the nature of the ultrasonic rangefinder itself, the range of its range cannot be correctly judged in the near part and the remote part.

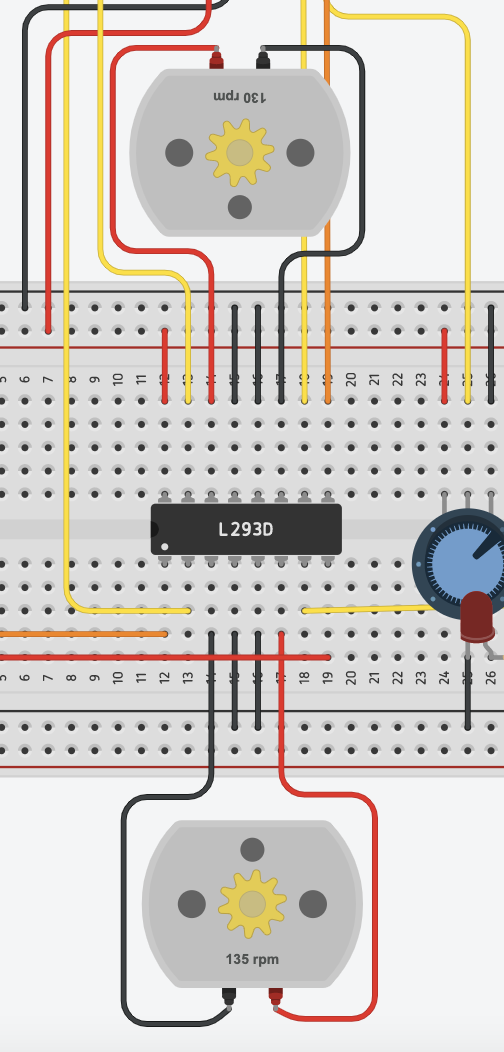
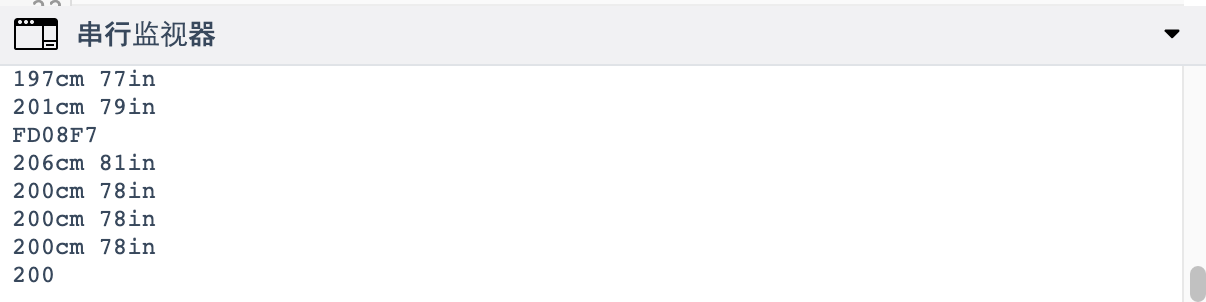
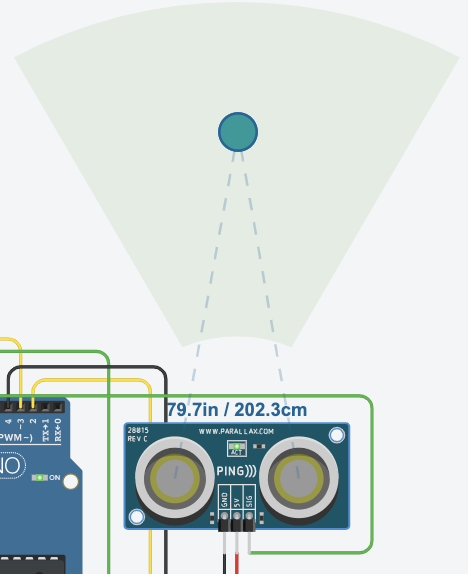
Therefore, to prevent normal fast forwarding without obstacles ahead, when the distance cannot be detected, it is always judged as fast forwarding.

Considering that the problem of being unavailable in very close situations is only 2cm, this problem can be prevented by retaining these two centimeters through the design of the hardware.

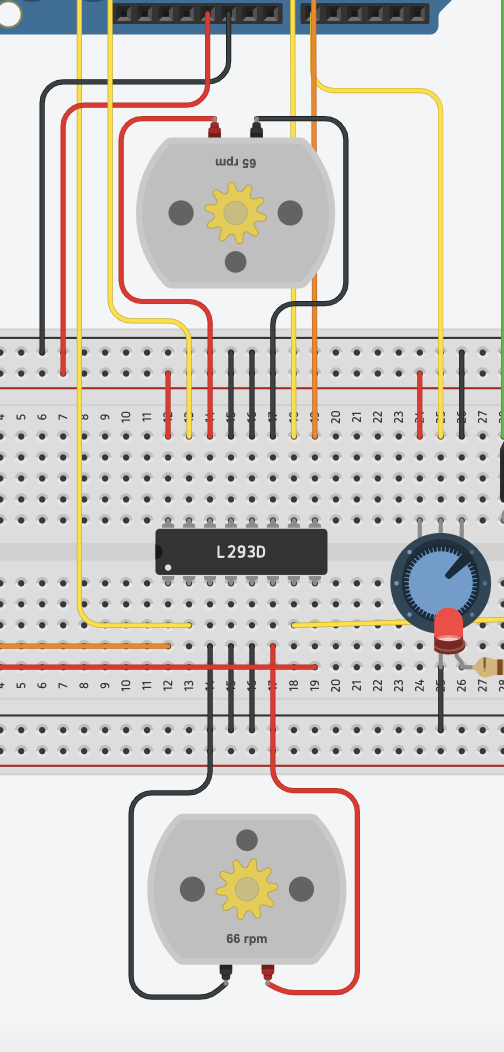
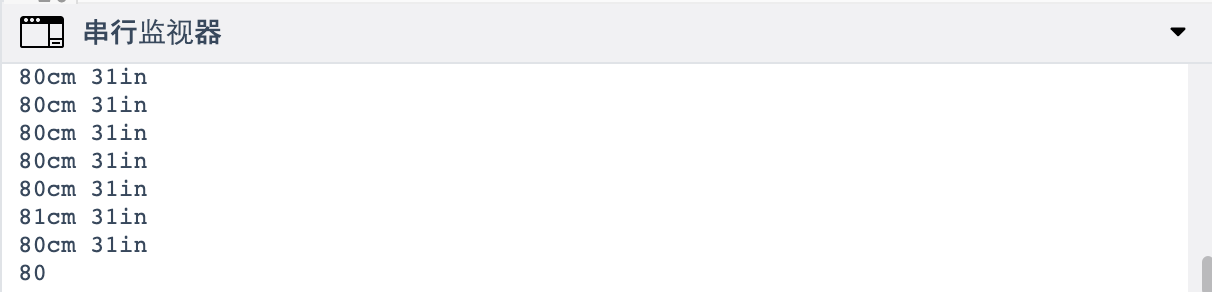
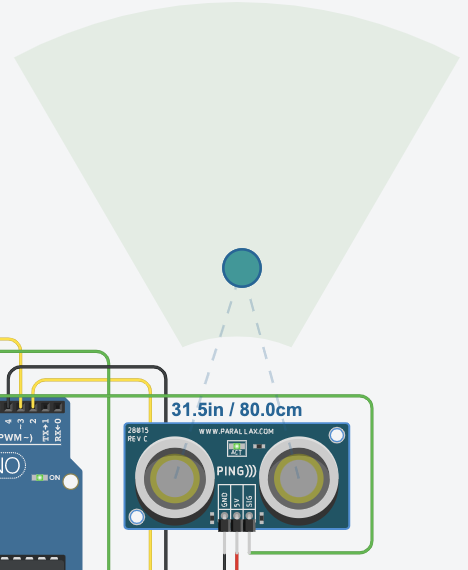
* 1. **Results**

In this part, the experimental results will be explained in detail. When we click to start the simulation, the ultrasonic rangefinder starts to detect the distance and displays the result in the serial monitor.

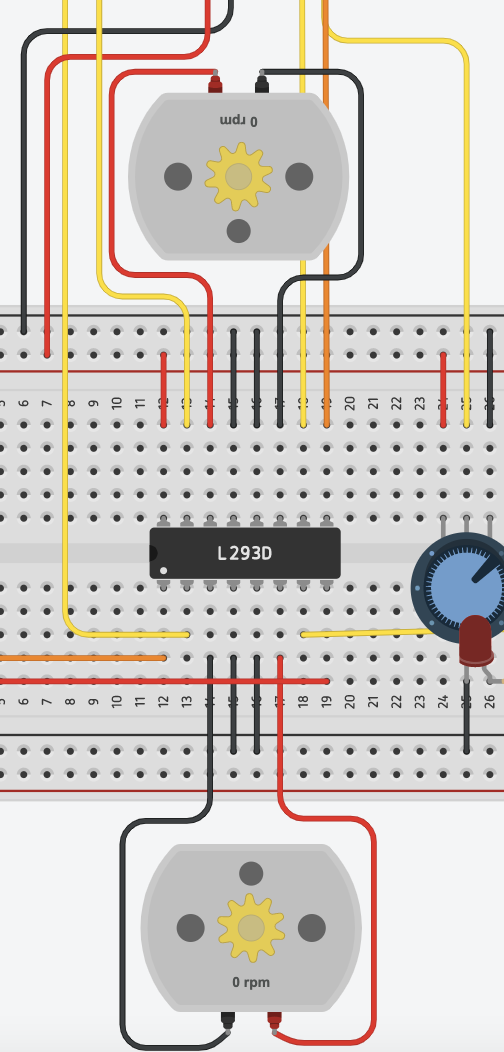
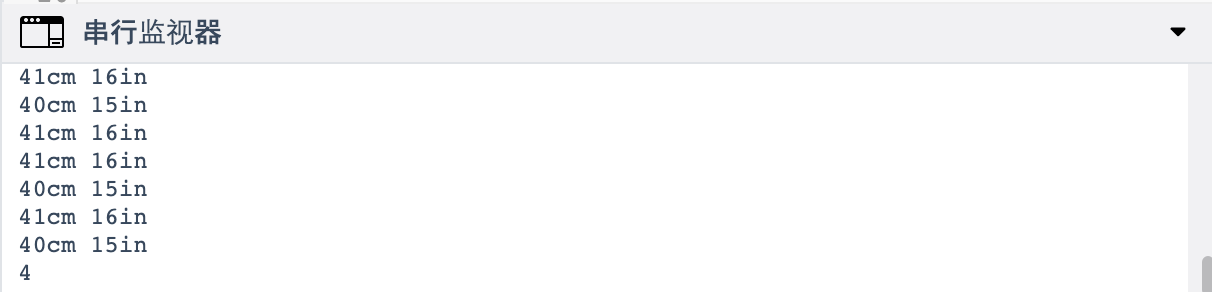
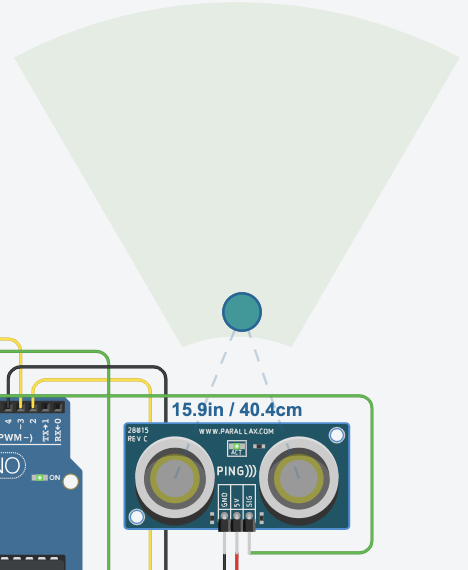
1. First, we set 200cm in the picture. When we press "1" on the IR remote controller, the hexadecimal number represented by "1" is displayed in the serial monitor, and the motor starts to rotate. Because "1" represents fast forward, the motor speed is about 120rpm.



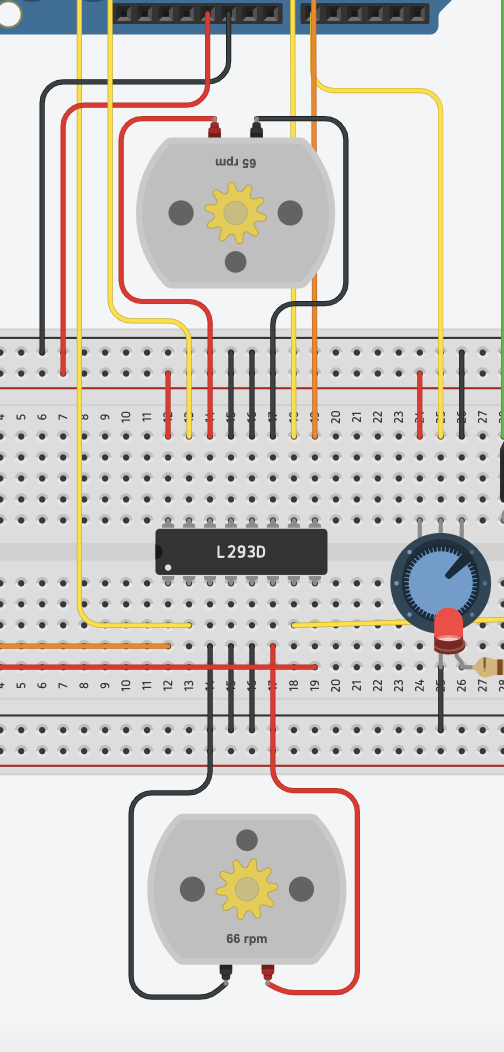
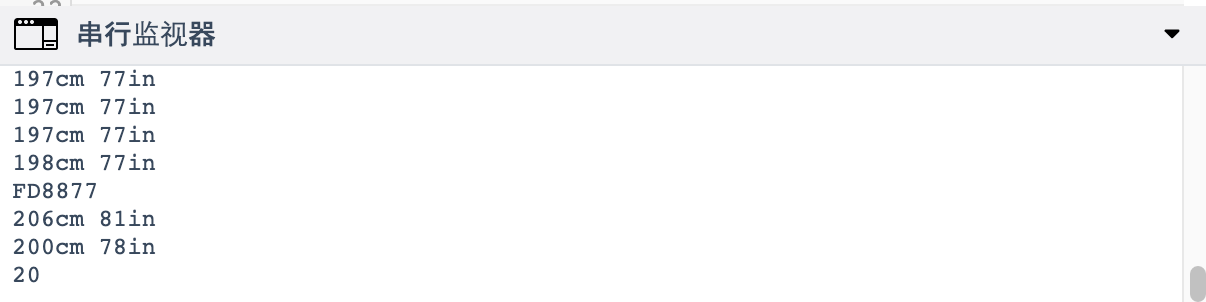
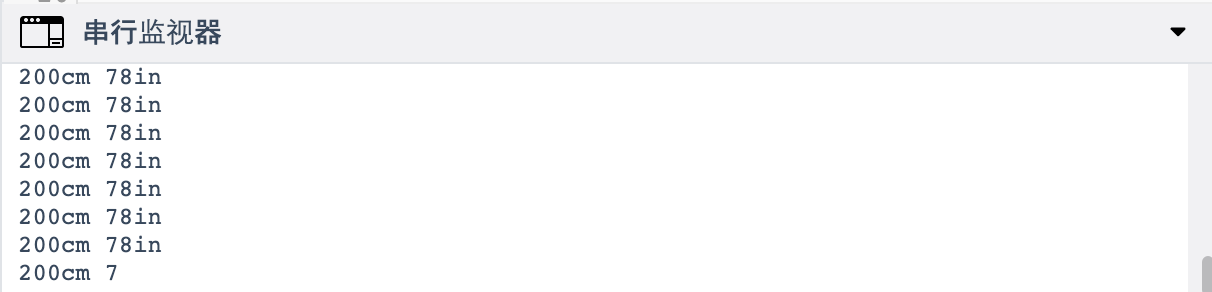
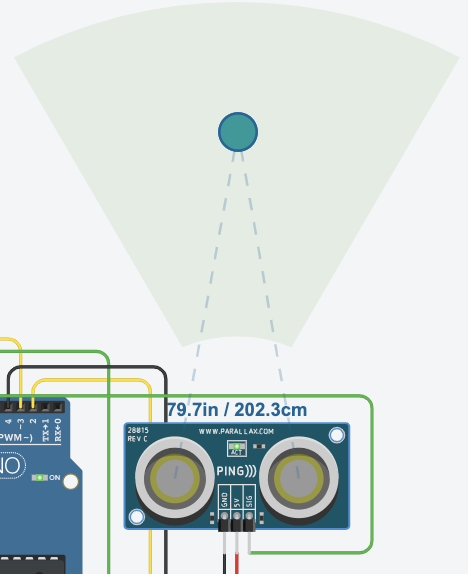
At this time, the distance is changed to 80cm, the motor speed is reduced to about 60rpm, and the LED becomes brighter.



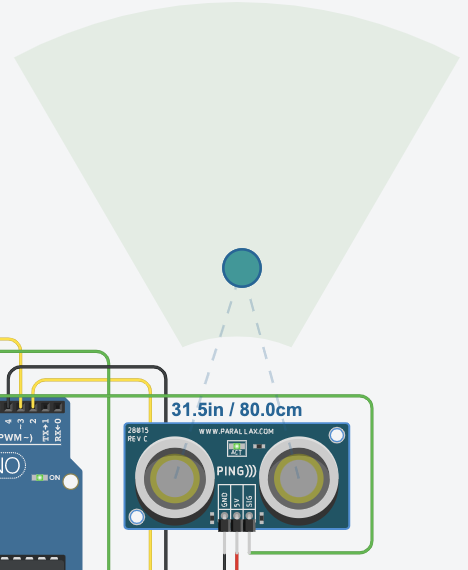
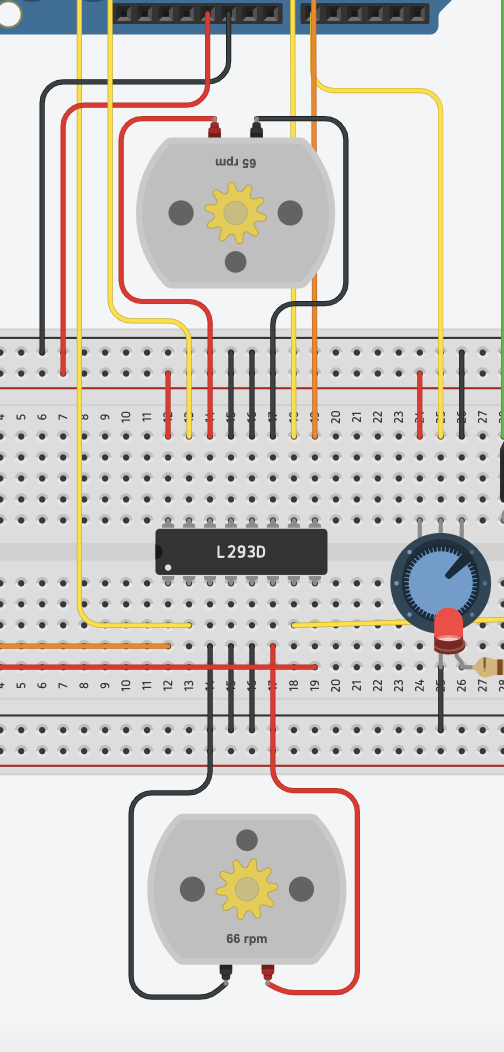
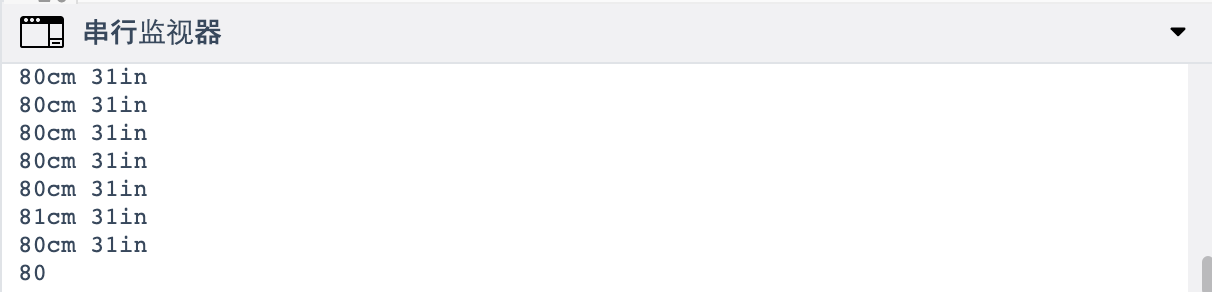
Change the distance to 40cm again, the motor stops rotating, and the LED goes out.



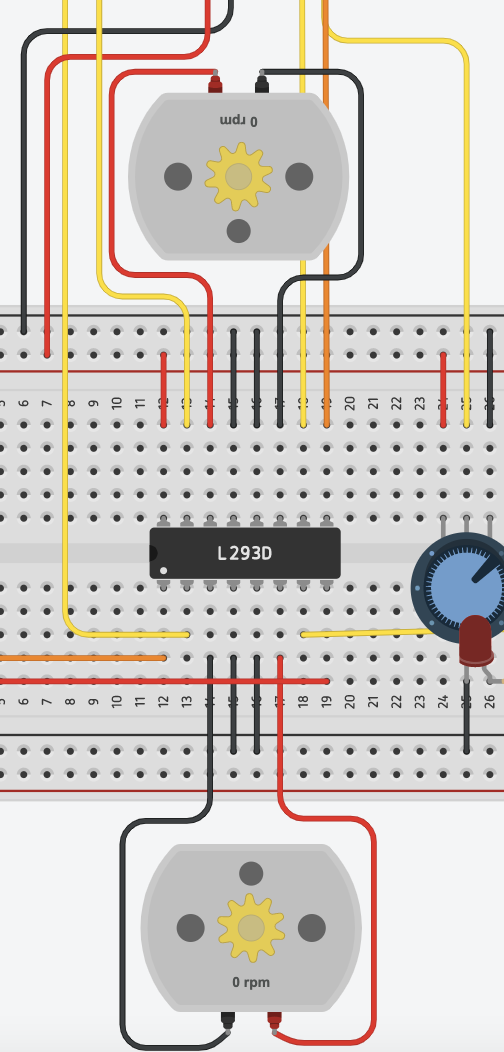
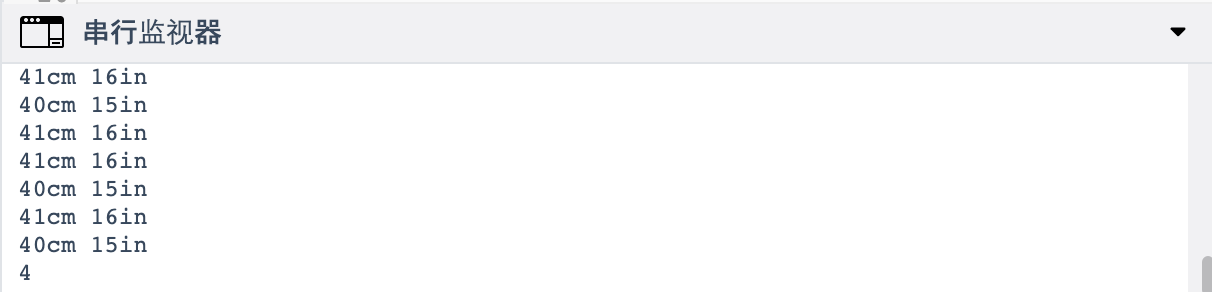
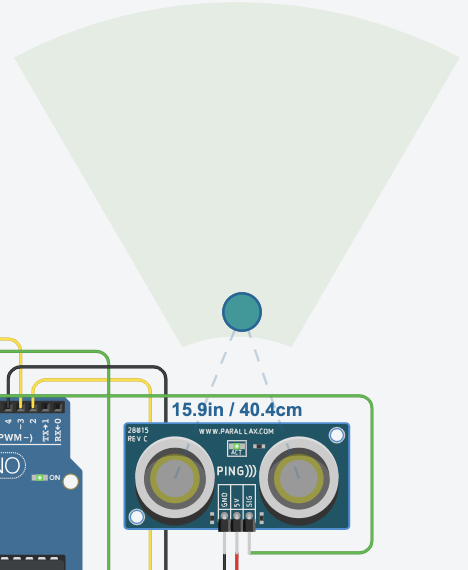
(2) At this time, adjust the distance back to 200cm and press "2" on the IR remote controller, the hexadecimal number represented by "2" is displayed on the serial monitor, and the motor rotates slowly at a speed of about 60rpm.



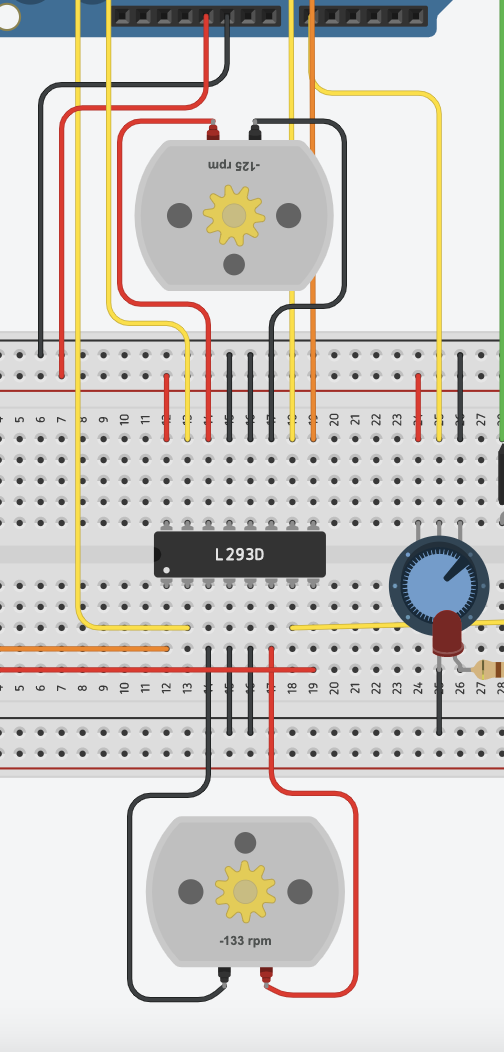
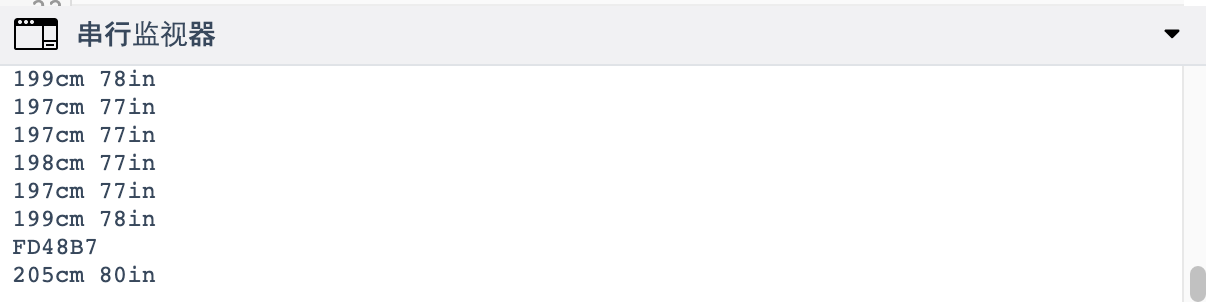
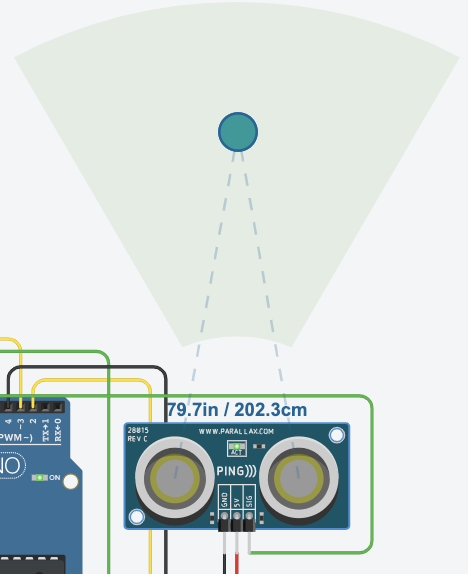
At this time, the distance is changed to 80cm, and the motor speed remains unchanged.

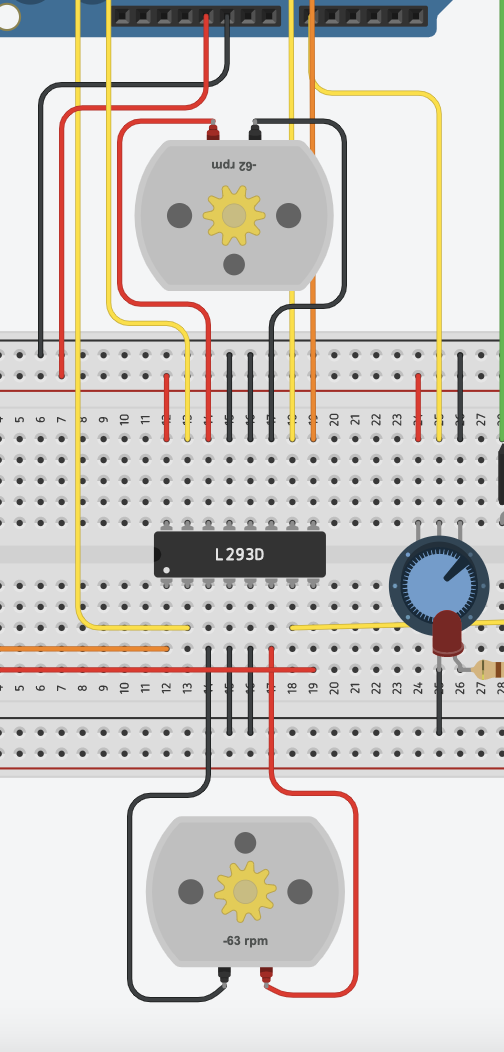
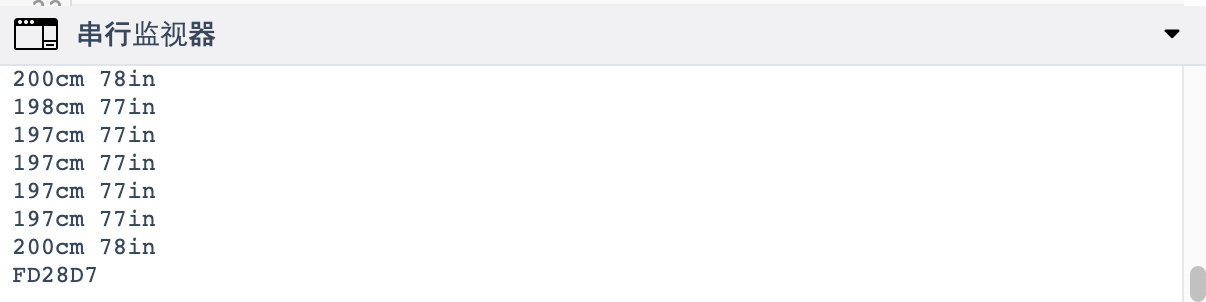
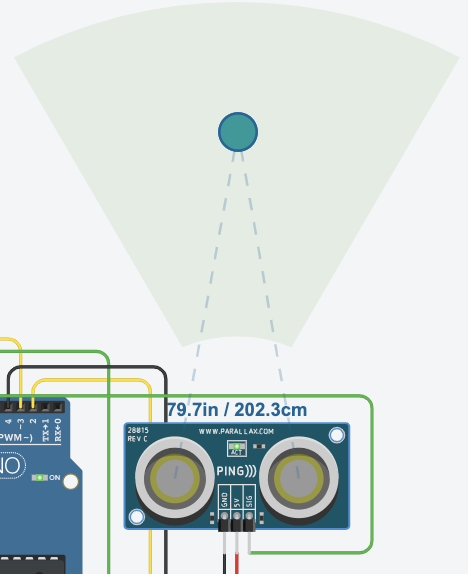
Change the distance to 40cm again, and the motor stops rotating.



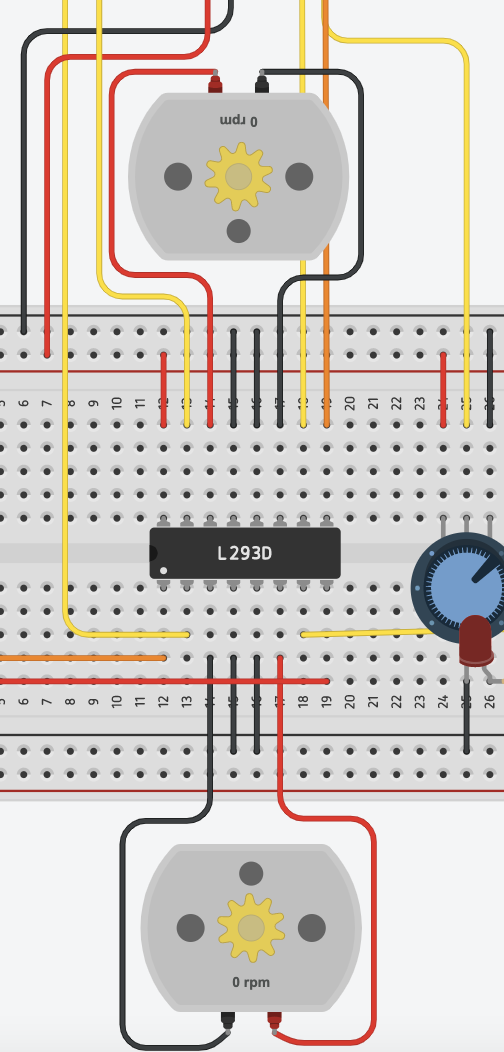
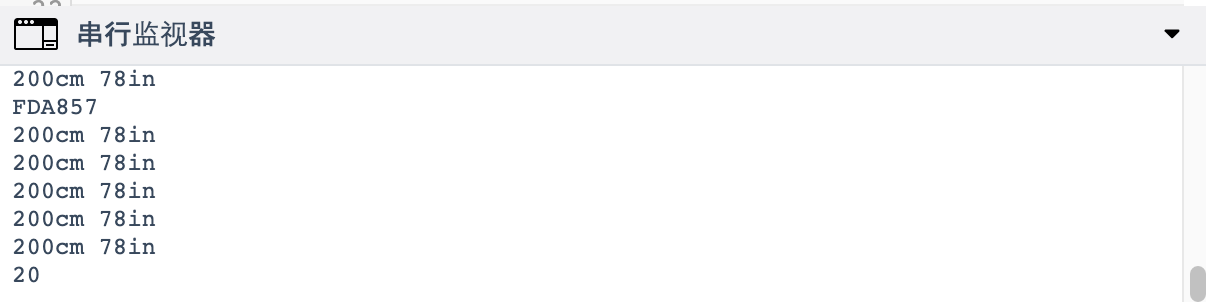
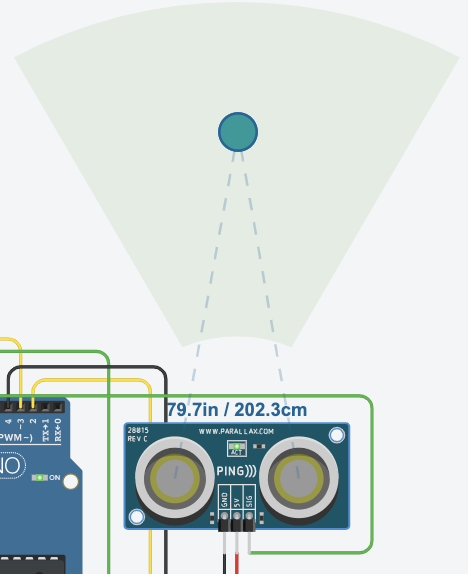
1. At this time, adjust the distance back to 200cm and press "3" on the IR remote controller, the serial monitor displays the hexadecimal number represented by "3" and the motor starts to rotate. Because "3" represents fast retreat, the motor speed is about -120rpm. At this time, the distance is changed to 80cm, and the motor speed is reduced to about -60rpm. Change the distance to 40cm again, and the motor stops rotating.



(4) At this time, adjust the distance back to 200cm and press "4" on the IR remote controller, the serial monitor displays the hexadecimal number represented by "4", and the motor rotates slowly in the reverse direction, the speed is- Around 60rpm. At this time, the distance is changed to 80cm, and the motor speed remains unchanged. Change the distance to 40cm again, and the motor stops rotating.



1. At any speed, when we press "5" on the IR remote controller, the serial monitor displays the hexadecimal number represented by "5" and the motor stops.

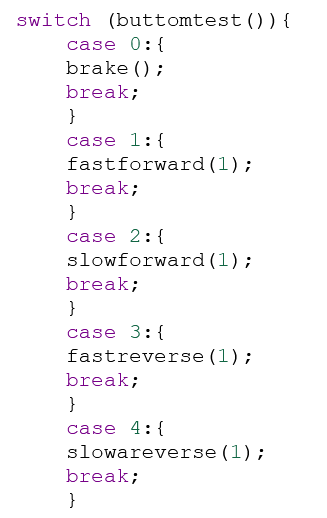


* 1. **Discussion**

With the joint efforts of the team members, we have realized the operation of the smart car. In this part, we will evaluate our experimental results from two aspects of basic function and advanced function. The evaluation will be mainly analyzed from the significance and reliability. By considering some errors in the experiment, we will seek possible alternatives to further improve the smart car.

**Basic function part:**

According to the basic function requirements, we use two DC motors, one L239D and one arduino uno board to control the smart car, so that the car has five functions, including fast forward, slow forward, fast reverse, slow reverse and brake. Through experiments, these basic functions have been fully realized.



In order to verify that the basic function can be performed, we make five buttons to perform the corresponding operations, writing the code shown above. Through the test, we found that the instructions given by each button can be executed, which proves the complete implementation of the basic function. The car reacts almost at the moment of receiving the button instruction, so that the smart car has a high efficiency. In this case, we can further design the advanced function.

**Advanced function part:**

1. **Remote control**

For the button must be directly connected to the smart car, it is unrealistic to use the button to control the car's action. Therefore, it is necessary for us to use remote control to realize the control of the car, which can also make the car operation more convenient and efficient. We connect each signal sent by the button on the remote control to the receiver to a basic function so that we can control the smart car with five buttons on the remote control. After testing, the smart car can react almost at the same time as the remote control sends the signal, which shows that our remote control worked sand the signal transmission efficiency is very high.

1. **Ultrasonic sensor**

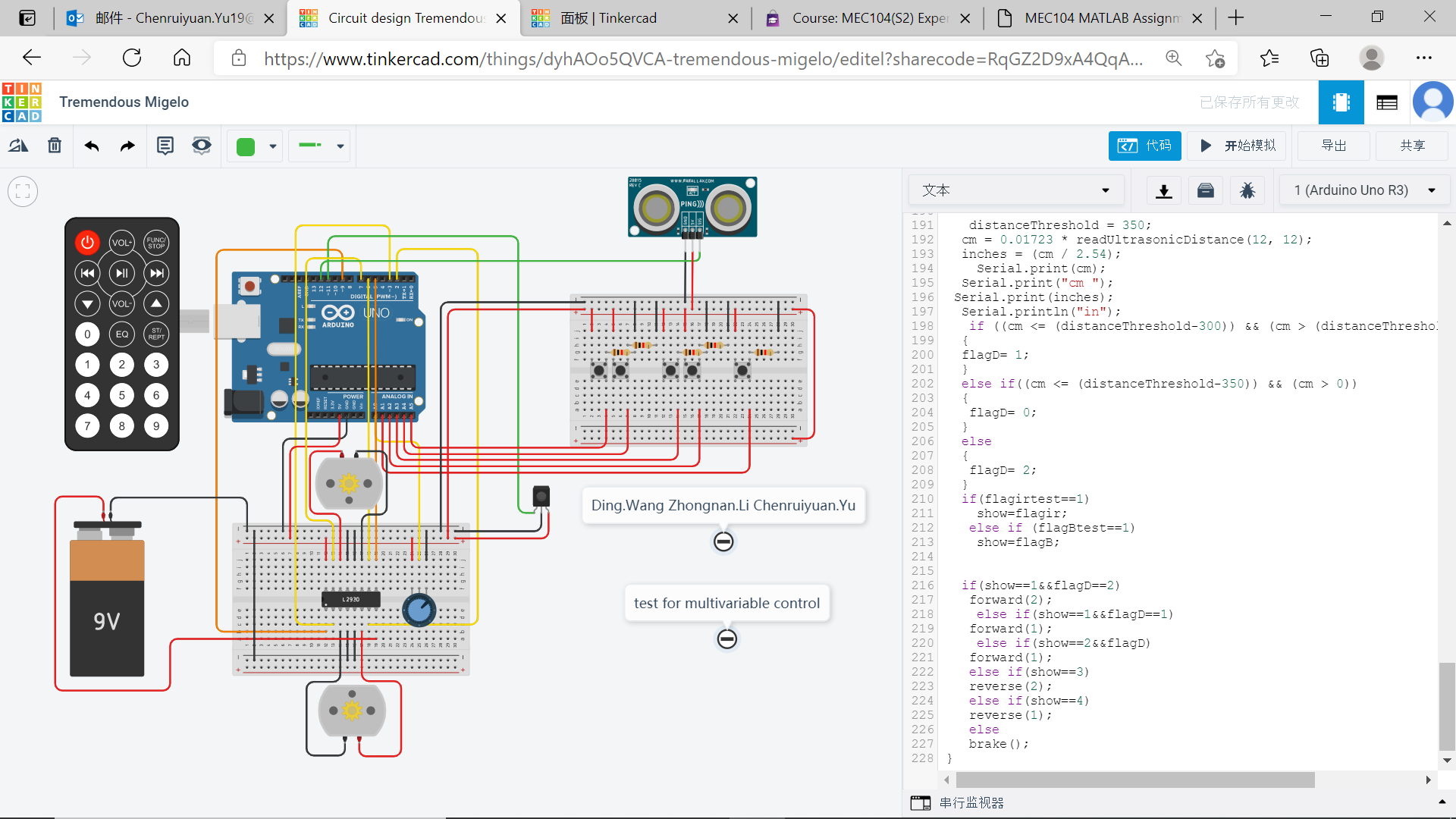
In order to ensure the safety of the car, it is necessary to add an ultrasonic sensor to control the speed by measuring the distance between the object in front of the car. We designed a program that if the distance of ultrasonic detection is more than 100 cm, the car runs at normal speed, if the distance is between 50 cm and 100 cm, the car slows down, finally if the distance is less than 50 cm, the car stops moving. It is worth mentioning that the priority of remote control is higher than that of ultrasonic, for instance, when the distance is less than 100 cm, if we still control the normal speed of the car, the smart car will not slow down. Although the reading value of the ultrasonic sensor in the serial monitor is slightly deviated from the actual value, it generally meets the requirement, in which case security can be highly guaranteed.

* 1. **Further study**

Initially, we implemented the basic function and the advanced function separately. Taking into account the special situation when the IR signal cannot be received, we hope to keep the operation of the physical button as a backup.

Therefore, we use five analog signal ports as inputs to record the button state, and then judge the speed of the smart car based on the state, and then implement button control.

In this process, the realization of multi-variable control requires more complex logic judgments, such as when IR conflicts with buttons.

In the simulation, the program we constructed initially solved the control conflict and the priority of different elements, but when one of the control elements made the same signal as the previous one, it could not be executed. Considering these flaws, we only present this part as a further study to reflect our efforts. These contents will be reflected in subsequent practice.

1. **Conclusion**

In short, the experiment has met all basic and advanced requirements and achieved a complete success.

We know the working principle of the car forward and backward at different speeds, how to stop, and the working principle of remote control and ultrasound.

Apart from the mastery of ARDUINO programming and the use of different modules, we have benefited a lot from the consideration of considering different elements for control in a complex system and the consideration of actual implementation.

Many contents of this project are universal and will be able to help us in the application of control and other contents.

1. **Author list and contributions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NO.** | **Name** | **Contribution description** | **Percentage (%)** | **signature** |
| **1** | Zhongnan Li 1931254 | IR test programming  Part of debugging  Several parts of report | **30** | **Zhongnan.Li** |
| **2** | Chenruiyuan Yu 1928118 | Basic function programming, functions integration and debugging process.  Report modification | **40** | **Chenruiyuan.Yu** |
| **3** | Ding Wang 1927804 | Ultrasonic distance measurement programming  Part of debugging  Several parts of report | **30** | **Ding.Wang** |

1. **Reference**
2. S. Karnouskos, "Self-Driving Car Acceptance and the Role of Ethics," *IEEE Transactions on Engineering Management*, vol. 67, no. 2, pp. 252-265, 2020, doi: 10.1109/tem.2018.2877307.
3. Available: <https://www.tinkercad.com/things/d0JjW5y7GmX>
4. Available: <https://www.tinkercad.com/things/3ntBpicMgVs>
5. HOW TO SET UP AN IR REMOTE AND RECEIVER ON AN ARDUINO Available: <https://www.circuitbasics.com/arduino-ir-remote-receivertutorial/>
6. Chen Y. Q. (2021)’ Week 5 Smart Car & Sustainability 1’, MEC104: Experimental, Computer Skills and Sustainability. Xi’an Jiaotong-Liverpool University, unpublished
7. **Appendix**

#include <IRremote.h>

#define in2 2

#define in1 3

#define st12 9

#define st34 5

#define in3 6

#define in4 7

#define led 4

int RECV\_PIN = 11; //define the pin of RECV\_PIN 11

IRrecv irrecv(RECV\_PIN); //define RECV\_PIN as infrared receiver

decode\_results results; //define variable results to save the result of IR reciever

int flagD;

int oldflagD;

int distanceThreshold=0;

int cm=0;

int inches=0;

long readUltrasonicDistance(int triggerPin, int echoPin)//get distance

{

pinMode(triggerPin, OUTPUT);

digitalWrite(triggerPin, LOW);

delayMicroseconds(2);

digitalWrite(triggerPin, HIGH);

delayMicroseconds(10);

digitalWrite(triggerPin, LOW);

pinMode(echoPin, INPUT);

return pulseIn(echoPin, HIGH);

}

void setup()

{

Serial.begin(9600);

Serial.println("Enabling IRin");

irrecv.enableIRIn();

Serial.println("Enabled IRin");

pinMode(st12, OUTPUT);

pinMode(in2, OUTPUT);

pinMode(in1, OUTPUT);

pinMode(st34, OUTPUT);

pinMode(in4, OUTPUT);

pinMode(in3, OUTPUT);

pinMode(led, OUTPUT);

}

void forward (int rate)

{

digitalWrite(led,LOW);

digitalWrite(st12, LOW);

digitalWrite(in2, HIGH);

digitalWrite(in1, LOW);

digitalWrite(st34, LOW);

digitalWrite(in4, HIGH);

digitalWrite(in3, LOW);

analogWrite(st12, rate);

analogWrite(st34, rate);

delay(100);

}

void forwardLED (int rate)

{

digitalWrite(led,HIGH);

digitalWrite(st12, LOW);

digitalWrite(in2, HIGH);

digitalWrite(in1, LOW);

digitalWrite(st34, LOW);

digitalWrite(in4, HIGH);

digitalWrite(in3, LOW);

analogWrite(st12, rate);

analogWrite(st34, rate);

delay(100);

}

void reverse (int rate)

{

digitalWrite(led,LOW);

digitalWrite(st12, LOW);

digitalWrite(in2, LOW);

digitalWrite(in1, HIGH);

analogWrite(st12, rate);

digitalWrite(st34, LOW);

digitalWrite(in4, LOW);

digitalWrite(in3, HIGH);

analogWrite(st34, rate);

delay(100);

}

void brake ()

{

digitalWrite(led,LOW);

digitalWrite(st12, LOW);

digitalWrite(in1, LOW);

digitalWrite(in2, LOW);

analogWrite(st12, HIGH);

digitalWrite(st34, LOW);

digitalWrite(in3, LOW);

digitalWrite(in4, LOW);

analogWrite(st34, HIGH);

}

void loop() {

distanceThreshold = 350;

cm = 2\*0.01723 \* readUltrasonicDistance(12, 12);

inches = (cm / 2.54);

Serial.print(cm);//show distance;

Serial.print("cm ");

Serial.print(inches);

Serial.println("in");

if ((cm <= (distanceThreshold-250)) && (cm > (distanceThreshold-300)))

flagD= 1;

else if((cm <= (distanceThreshold-300)) && (cm > (distanceThreshold-350)))

flagD= 0;

else

flagD= 2;

if (irrecv.decode(&results)||(flagD!=oldflagD)) {//ir control;

Serial.println(results.value, HEX);

switch(results.value){

case 0xFD08F7:

{

if(flagD==2)

forward (2);

if(flagD==1)

{

forwardLED(1);

}

if(!flagD)

brake();

break;}

case 0xFD8877:{

if(flagD)

forward (1);

else

brake();

break;}

case 0xFD48B7:

reverse (2);

break;

case 0xFD28D7:

reverse (1);

break;

case 0xFDA857:

brake();

break;

}

irrecv.resume();

oldflagD=flagD;

}

}