AD176

ESP32 WiFi Camera Mecanum Smart Robot Car Kit

V2.0

Preface

Our Company

KUONGSHUN Electronic Company is a supplier and manufacturer of electronic components. We are committed to providing development boards and starter kits for Arduino, Raspberry Pi, smart robot cars, and 3D printers. As a high-tech enterprise integrating scientific research, design, production, maintenance, and sales, we uphold professional standards and mature technologies in the field of automation. With these strengths, we have rapidly emerged in the international trade market.

Relying on technology for development and continuously providing users with high-tech products is our constant pursuit. We fully introduce advanced foreign technologies to enhance the value of our products.

The company has earned praise from users for providing first-class products and superb technical services, and has now become the preferred choice of both domestic and international buyers.

Official Website: https://www.kuongshun.com

Our Tutorial

This course and learning kit is designed for children and teens ages 8 and up to learn about Arduino-compatible boards, actuators, sensors and related components, and programming.

This tutorial aims to introduce learners to how to program the car and how to control the car once it has been programmed. The car is equipped with Wi-Fi control, video transmission, and mode switching functions. As a common STEM educational tool, it not only fosters learners' hands-on abilities but also helps them understand basic programming concepts and logic.

Customer Service

As a continuously and fast-growing technology company, we keep doing our best to offer you excellent products and quality service to meet your expectations. You can reach out to us by simply dropping a line at support@kuongshun.cn. We look forward to hearing from you, and any of your critical comments or suggestions would be very valuable to us.

Any problems or questions you have with our products will be promptly answered by our experienced engineers within 12 hours (24 hours during holidays).

We pursue the policy of progress, truth, rigor, and unity, fostering innovation, focusing on technology as the core, committing to quality, and making customer satisfaction our top priority. We are dedicated to providing you with the most cost-effective high-tech products and attentive service.

Packing List

Content

Preparation 1
Lesson 1 Get The Car Moving 4
Lesson 2 Servo Pan-Tilt 14
Lesson 3 Ultrasonic Sensor Module 22
Lesson 4 Three Infrared Line Tracing 31
Lesson 5 WIFI Camera Multi-Function Program for Web 40
Lesson 6 WIFI Camera Multi-Function Program for APP 55
Frequently Asked Questions 74

Preparation

Before proceeding with this tutorial, please make sure you have completed the following five steps:

1.Complete the model assembly as described in "1_Assembly Guide," as shown in the image below: (If you purchased the version without batteries, please prepare or purchase two 18650 batteries separately.)

2.Install the Arduino software mentioned in "2_Programming Preparation," as shown in the image below.

3.Install the CH340 driver mentioned in "2_Programming Preparation," as shown in the image below.

4. Install the ESP32 board mentioned in "2_Programming Preparation," as shown in the image below.

Note: Please install version 2.0.12 (do not install versions above 3.0, as this may cause errors when uploading the program).

5.Add the library file:

If any step has not been completed, refer to "1_Assembly_Guide or 2_Programming_Preparation".

Lesson 1 Get The Car Moving

In this project, we will use Arduino compatible board combined with L293D extension board to realize the walking function of the car. This car adopts a four-wheeled Mecanum wheel structure, which can realize multi-directional translation and steering.

The Mecanum wheel is a type of non-traditional wheel that features a series of rubber rollers mounted at an angle around the entire circumference of the wheel hub. These rollers' rotation axes are typically set at a 45-degree angle relative to both the wheel plane and the axle. Each Mecanum wheel operates as an independent non-steering drive wheel, with its own power system. When the wheel rotates, it generates a force perpendicular to the roller axis, which can be split into longitudinal and lateral components relative to the vehicle. When all four wheels move simultaneously, some of the forces are canceled out, allowing the vehicle to move in the desired direction without changing its orientation, enabling lateral and diagonal motion.

We'll learn how to upload a program to a car first, and we'll learn how Mecanum wheels work later:

①Open the '1_Auto_Move.ino' program in 'KUONGSHUN AD176 ESP32 WiFi Camera Mecanum 4WD Smart Robot Car\3_Code_and_Tutorial\1_Auto_Move'.

②Connect the Arduino board to your computer with a TYPE-C cable.

③ Select your Board in Tools > Board > Arduino AVR Board > Arduino UNO.

④Select the Port: Tools > Port > COMx(The COM port number of each computer may be different, and the port number that appears is your port number) (If you do not see COM ports other than COM1 in Arduino IDE, you need to refer to the installation of CH340 driver in "2_Programming_Preparation" and install it.)

⑤Unplug the serial port transmission cable between the Arduino board and the ESP32-CAM board.(This cable will not be connected back during all uploading procedures, until finally all procedures are uploaded and the car is controlled by WIFI.)

The Arduino and ESP32-CAM boards communicate through the RX and TX pins, which are also used by the computer to upload programs. If the serial connection between the Arduino and ESP32-CAM boards is not disconnected, the computer will be unable to upload the program due to the serial port being occupied.

⑥Upload the program to the UNO controller board.

The picture above shows that it is uploaded successfully.

⑦After uploading the code successfully. Disconnect the data line, put the car on the ground, and load 2 18650 batteries. (If you buy the battery-free version, please prepare two 18650 batteries.) Then turn on the switch.

At this point, the car will go forward - backward - go left - go right - go up left - go down right - go down left - go up right - rotate clockwise - rotate counterclockwise - stop, and these 11 states will run for 2 seconds in turn and cycle.

Let's see the Mecanum wheel. Here are the wheels in two directions:

Almost all 4WD Mecanum wheels are installed with this mounting:

Motor Drive and Motion Principle:

In the code, PWM pins 5,6 control the power (speed) of the wheel, and then through the 74HC595 chip pins 2,7,8,4 control the rotation direction of each motor.

```
// PWM control pin
#define PWM1_PIN        5
#define PWM2_PIN        6
// 74HCT595N Chip pins
#define SHCP_PIN        2
#define EN_PIN          7
#define DATA_PIN        8
#define STCP_PIN        4
```

We can see that the 74HC595D signal port is connected to only four, and a motor generally requires two signal ports to control the direction. Let's learn how the 74HC595D works.

Below is the chip wiring diagram of 74HC595D.

In the core operation program of the motor, the shiftOut function is the main one. The high and low levels of each pin of the motor are controlled by decimal numbers converted from 8-bit binary numbers in the range 0-255.

The shiftOut function has a total of four parameters.The first three parameters are defined and configured at the beginning, and we only need to change the value of Dir. At this time, the system converts the decimal number into an 8-bit binary number to control the high and low levels of the motor.

```
void Motor(int Dir, int Speed)
{
    digitalWrite(EN_PIN, LOW);
    analogWrite(PWM1_PIN, Speed);
    analogWrite(PWM2_PIN, Speed);
    digitalWrite(STCP_PIN, LOW);
    shiftOut(DATA_PIN, SHCP_PIN, MSBFIRST, Dir);
    digitalWrite(STCP_PIN, HIGH);
}
```

For example, if you want a car to move forward, you have to keep all four wheels moving forward. The corresponding 8-bit code is 10100011, which converts to the decimal value 163. If you want the car to go backwards, you have to keep all four wheels turning backwards. The corresponding 8-bit code is 01011100, which converts to the decimal value 92.

Coding Value Table Corresponding to Different Motion States (When M1+ is 1 and M1- is 0, M1 advances, When M1+ is 0, M1- is 1, M1 backs off, and the other motors are similar)										
State Of The Car	M1+	M1-	M2+	M2-	M4-	M3-	M3+	M4+	8 bit binary	Decimal system
Move Forward	1	0	1	0	0	0	1	1	10100011	163
Move Backward	0	1	0	1	1	1	0	0	01011100	92
Move Left	0	1	1	0	1	0	1	0	01101010	106
Move Right	1	0	0	1	0	1	0	1	10010101	149
Move Upper Left	0	0	1	0	0	0	1	0	00100010	34
Move Lower Left	0	1	0	0	1	0	0	0	01001000	72
Move Upper Right	1	0	0	0	0	0	0	1	10000001	129
Move Lower Right	0	0	0	1	0	1	0	0	00010100	20
Stop	0	0	0	0	0	0	0	0	00000000	0
Contrarotate	0	1	0	1	0	0	1	1	01010011	83
Clockwise	1	0	1	0	1	1	0	0	10101100	172

Set a variable for each motion state that stores the decimal code corresponding to the car's motion state:

```
const int Forward        = 163;
const int Backward       = 92;
const int Turn_Left      = 106;
const int Turn_Right     = 149;
const int Top_Left       = 34;
const int Bottom_Left    = 72;
const int Top_Right      = 129;
const int Bottom_Right   = 20;
const int Stop           = 0;
const int Contrarotate   = 83;
const int Clockwise      = 172;
```

Lesson 2 Servo Pan-Tilt

In this project, we will learn the use of steering gear. It also tests and corrects the Angle of the servo motor. The first thing to do is to upload the servo Angle test program to the Arduino board:

①Open the '2_Servo_Angle.ino' program in 'KUONGSHUN AD176 ESP32 WiFi Camera Mecanum 4WD Smart Robot Car\3_Code_and_Tutorial\2_Servo_Angle'.

② Connect the Arduino board to the computer with TYPE-C data cable.

③ Select your control panel: Tools > Board > Arduino AVR Board > Arduino UNO.

④Select the Port: Tools > Port > COMx(The COM port number of each computer may be different, and the port number that appears is your port number) (If you do not see COM ports other than COM1 in Arduino IDE, you need to refer to the installation of CH340 driver in "2_Programming_Preparation" and install it.)

⑤Unplug the serial port transmission cable between the Arduino board and the ESP32-CAM board.(This cable will not be connected back during all uploading procedures, until finally all procedures are uploaded and the car is controlled by WIFI.)

⑥Upload the program to the UNO controller board.

The picture above shows that it is uploaded successfully.

⑦After uploading the code successfully. Disconnect the data line, put the car on the ground, and load 2 18650 batteries. (If you buy the battery-free version, please prepare two 18650 batteries.) Then turn on the switch.The servo motor of the locomotive will turn to the 90 degree position, stay for one second and then swing up and down. We specify that the orientation is straight forward at 90 degrees, as shown below:

The amplitude of the back-and-forth swing is shown below:

If the swing amplitude of the car is not consistent with the figure above, it shows that your servo placement is not very accurate. We need to press the reset button of the Arduino board to bring the servo back to 90 degrees. Then unplug the data cable to stop the steering gear. At this time, separate the servomotor rocker arm from the ESP32-CAM support bracket as shown in the figure.Straightening the whole structure and then buckle back.

The servo gimbal movement at this point is as follows:

Principle Introduction:

SG90 Servo Motor

Classification: 180 steering gear.

Pin Definitions::brown line --GND, red line --5V, orange line -- signal.

How the Servo Motor Works:

The signal modulation chip in the servo motor receives the signal from the control board, and then the servo motor will obtain the basic DC voltage. There is also a reference circuit inside the servo motor, which will generate a standard voltage. These two voltages are compared with each other and then the difference is output. The motor chip will then receive the difference and decide on the speed, direction, and angel. The servo motor stops when there is no difference between the two voltages.

Servo Motor Control Principle:

To control the servo rotation, it is necessary to control the time pulse at about 20ms and the high level pulse width at about 0.5ms to 2.5ms, which is consistent with the angular limit of the servo.

Taking 180 Angle servo as an example, the corresponding control relationship is as follows:

0.5ms	0 degrees
1.0ms	45 degrees
1.5ms	90 degrees
2.0ms	135 degrees
2.5ms	180 degrees

At the marked position of the program in the following figure, we can modify the initial Angle and rotation range of the servo. Follow the same steps to upload the program once the changes are made. Then we can see the result of the change.

Lesson 3 Ultrasonic Sensor Module

In this project, we will learn how to use the ultrasound module and how to use the serial port monitor to view the distance data returned by the ultrasound. Before we do this, we need to upload the program to the car:

①Open the '3_Ultrasonic.ino' program in 'KUONGSHUN AD176 ESP32 WiFi Camera Mecanum 4WD Smart Robot Car\3_Code_and_Tutorial\3_Ultrasonic'.

② Connect the Arduino board to the computer with TYPE-C data cable.

③ Select your control board: Tools > Board > Arduino AVR Board > Arduino UNO.

④Select the Port: Tools > Port > COMx(The COM port number of each computer may be different, and the port number that appears is your port number) (If you do not see COM ports other than COM1 in Arduino IDE, you need to refer to the installation of CH340 driver in "2_Programming_Preparation" and install it.)

⑤Unplug the serial port transmission cable between the Arduino board and the ESP32-CAM board.(This cable will not be connected back during all uploading procedures, until finally all procedures are uploaded and the car is controlled by WIFI.)

⑥Upload the program to the UNO controller board.

The picture above shows that it is uploaded successfully.

⑦ After uploading the code successfully. Open the serial port monitor and confirm that the baud rate is 9600. You can see the distance information of the ultrasonic wave appearing on the monitor.

Principle Introduction:

Module Features: Test distance, high precision module.

Applications: Obstacle avoidance, object testing distance, liquid testing, public safety, parking lot testing.

Main Technical Parameters:

(1) Use voltage: DC--5V

(2) Static current: less than 2mA

(3) Level output: The high-level output is approximately 5V, and the low-level output is approximately 0V

(4) Detection angle: no more than 15 degrees

(5) Detection distance: 2cm-450cm

(6) High precision: up to 0.2 cm

(7) Pin Description: VCC, trig (control pin), echo (receiver pin), GND

How Modules Work:

(1) Triggering the Measurement

Connect the TRIG pin to the controller's IO port. By outputting a high-level signal lasting at least 10 microseconds, the measurement process is initiated.

(2) Sending Ultrasonic Pulses

The module automatically emits 8 ultrasonic square wave pulses at a frequency of 40kHz and listens for any echo signals returned.

(3) Receiving Echo Signals

When an echo is detected, the module outputs a high-level pulse on the ECHO pin. The duration of this pulse corresponds to the time interval between the emission and reception of the ultrasonic wave. By measuring this pulse width, the module calculates the distance to the target.

$$\textbf{Distance} = \frac{\textbf{High Level Time} * \textbf{Sound Speed}(340M/S)}{2}$$

Program Logic:

3_Ultrasonic.ino

```
//www.kuongshun.com

float checkdistance()
{
    digitalWrite(12, LOW);
    delayMicroseconds(2);
    digitalWrite(12, HIGH);
    delayMicroseconds(10);
    digitalWrite(12, LOW);
    float distance = pulseIn(13, HIGH) / 58.00;
    delay(10);
    return distance;
}

void Ultrasonic_Sensor_Module()
{
    int Distance = 0;
    Distance = checkdistance();
    Serial.print("Distance:");
    Serial.print(Distance);
    Serial.println("CM");
    delay(100);
}
```

The program is used to detect the distance of an object using an ultrasonic sensor, with the Trig (trigger signal) connected to pin 12 and the Echo (echo signal) connected to pin 13.

Emit Ultrasonic Waves

Set the Trig pin to LOW and hold it for 2 microseconds to clear any potential residual signals from the previous transmission.

```
digitalWrite(12, LOW);
delayMicroseconds(2);
```

Set the Trig pin to HIGH and hold it for 10 microseconds to trigger the ultrasonic sensor to emit 8 ultrasonic pulses at 40kHz.

```
digitalWrite(12, HIGH);
delayMicroseconds(10);
```

Set the Trig pin back to LOW to complete the trigger signal.

```
digitalWrite(12, LOW);
```

At this point, we have triggered the ultrasonic pulse by sending a 10-microsecond pulse. When the ultrasonic wave is reflected back from an obstacle, the Echo pin's voltage level will go high. We measure the duration of the high-level pulse on the Echo pin (in microseconds).

Use pulseIn(13, HIGH) to measure the duration of the high-level pulse on the Echo pin (in microseconds). This duration represents the round-trip time of the ultrasonic wave from transmission to reception.

To convert the time to distance (in centimeters), use the following formula:

Distance = (Time × Speed of Sound) / 2

The speed of sound in air is approximately 343 meters per second (or 34300 centimeters per second). Since the time is measured in microseconds (1 second = 1,000,000 microseconds)

$$\textbf{Distance(cm)} = \frac{\textbf{Time(Microseconds)}}{\textbf{58}}$$

```
float distance = pulseIn(13, HIGH) / 58.00;
```

Finally, wait for 10 milliseconds to give the sensor a brief interval, preventing continuous triggers that could lead to measurement confusion.

```
delay(10);
```

Note: The ultrasonic sensor has a measurement range limitation, from 2 centimeters to 400 centimeters.

Lesson 4 Three Infrared Line Tracing

In this project, we will learn the use of three-way infrared line patrol module and how to make the car complete a complete line patrol. Start by uploading the program to the car.

① Open the '4_Line_Tracing.ino' program in 'KUONGSHUN AD176 ESP32 WiFi Camera Mecanum 4WD Smart Robot Car\3_Code_and_Tutorial\4_Line_Tracing'.

② Connect the Arduino board to the computer with TYPE-C data cable.

③ Select your control board: Tools > Board > Arduino AVR Board > Arduino UNO.

④Select the Port: Tools > Port > COMx(The COM port number of each computer may be different, and the port number that appears is your port number) (If you do not see COM ports other than COM1 in Arduino IDE, you need to refer to the installation of CH340 driver in "2_Programming_Preparation" and install it.)

⑤Unplug the serial port transmission cable between the Arduino board and the ESP32-CAM board.(This cable will not be connected back during all uploading procedures, until finally all procedures are uploaded and the car is controlled by WIFI.)

⑥Upload the program to the UNO controller board.

The picture above shows that it is uploaded successfully.

⑦After uploading the code successfully. Disconnect the data line, put the car on the ground, and load 2 18650 batteries. (If you buy the battery-free version, please prepare two 18650 batteries.) Then turn on the switch.We need to find a light or white surface, put black tape on it, and test whether the car can tell the difference between the black tape and the light background. The steps are as follows:

a. Attach the next section with black tape on the appropriate plane.

b. Turn on the battery box switch and place the car on the table. Aim the left infrared sensor at the black tape; the corresponding red indicator light will turn on. Then, move the left infrared sensor toward the adjacent white surface, and the red indicator light will turn off.(Note: If the red indicator light does not turn on when the sensor is aimed at the black tape, please rotate the potentiometer knob to adjust it until the red light lights up correctly when facing the black tape.)

c. The same method should be used to test the other two infrared sensors.

d. All three lights are off against a white background.

e. The three lights are on against the black line.

If the above steps are correct, you can make a map and start your line patrol.

Turn on the switch and the car will walk along the line.

The logic and code of the walk are as follows: the left test black to the left, the right test black to the right, only the middle test black when forward.

```
Left_Tra_Value = analogRead(LEFT_LINE_TRACJING);
Center_Tra_Value = analogRead(CENTER_LINE_TRACJING);
Right_Tra_Value = analogRead(right_LINE_TRACJING);
if (Left_Tra_Value > Black_Line && Center_Tra_Value <= Black_Line && Right_Tra_Value > Black_Line)
{
  Motor(Forward, 180);
}
else if (Left_Tra_Value < Black_Line && Center_Tra_Value < Black_Line && Right_Tra_Value < Black_Line)
{
  Motor(Forward, 130);
}
else if (Left_Tra_Value >= Black_Line && Center_Tra_Value >= Black_Line && Right_Tra_Value < Black_Line)
{
  Motor(Clockwise, 220);
}
else if (Left_Tra_Value >= Black_Line && Center_Tra_Value < Black_Line && Right_Tra_Value < Black_Line)
{
  Motor(Clockwise, 160);
}
else if (Left_Tra_Value < Black_Line && Center_Tra_Value < Black_Line && Right_Tra_Value >= Black_Line)
{
  Motor(Contrarotate, 160);
}
else if (Left_Tra_Value < Black_Line && Center_Tra_Value >= Black_Line && Right_Tra_Value >= Black_Line)
{
  Motor(Contrarotate, 220);
}
else if (Left_Tra_Value >= Black_Line && Center_Tra_Value >= Black_Line && Right_Tra_Value >= Black_Line)
{
  Motor(Forward, 130);
}
```

Three Infrared Line Tracing

The inspection sensor is located under the front direction of the bottom plate of the car. Each route of the patrol sensor consists of an infrared transmitter tube and an infrared receiver tube. The former is a light-emitting diode that can emit infrared light, and the latter is a photoresistor that is only responsible for receiving infrared light. Black surfaces have different light reflectance than white surfaces. Therefore, the intensity of the infrared reflected light received

by the car on the black road surface is different from that on the white road surface, and the resistance value changes accordingly. According to the principle of series resistance voltage division, the color of the road under the car can be inferred from the voltage of the sensor, so as to determine the motion path.

Tips:

(1) The bending part of the line should be as smooth as possible. If the turning radius is too small, the car is likely to overshoot the runway.

(2) Straight line tracking scenes can be made with black and white tape or any color paper to distinguish them from paths.

(3) In addition to line tracking, we can use our imagination to develop other programs, such as programs that confine a car to a certain area regardless of its motion.

Lesson 5 WIFI Camera Multi-Function Program for Web

In this project, we will use Arduino board combined with L293D extension board to realize the functions of walking, obstacle avoidance, line inspection and route of the car. The ESP32-CAM is used to realize WiFi connection and control the car through the serial port line. At the same time, ESP32-CAM also transmits the image of the front camera to the hotspot server in real time. The mobile phone input the hotspot can view the camera image of the car and control the walking of the car, as well as adjust the operation mode and the speed of the car.

Before presenting the multi-function program of the cart, we first upload the program to the cart. The car has two control boards, and we need to upload two programs to each board. Here are the steps to upload an Arduino board and program:

①Open the '5.1_Arduino_All_Web.ino' program in 'KUONGSHUN AD176 ESP32 WiFi Camera Mecanum 4WD Smart Robot Car\3_Code_and_Tutorial\5.1_Arduino_All_Web'.

② Connect the Arduino board to the computer with TYPE-C data cable.

③ Select your control board: Tools > Board > Arduino AVR Board > Arduino UNO.

④Select the Port: Tools > Port > COMx(The COM port number of each computer may be different, and the port number that appears is your port number) (If you do not see COM ports other than COM1 in Arduino IDE, you need to refer to the installation of CH340 driver in "2_Programming_Preparation" and install it.)

⑤Unplug the serial port transmission cable between the Arduino board and the ESP32-CAM board.(This cable will not be connected back during all uploading procedures, until finally all procedures are uploaded and the car is controlled by WIFI.)

⑥Upload the program to the UNO controller board.

The picture above shows that it is uploaded successfully.

⑦After uploading the code successfully.We also need to upload the program to the **ESP32-CAM board** of the locomotive.

Open the '5.2_ESP32Cam_WIFI_Web.ino' program in 'KUONGSHUN AD176 ESP32 WiFi Camera Mecanum 4WD Smart Robot Car\3_Code_and_Tutorial\5.2_ESP32Cam_WIFI_Web'.

⑧Connect the ESP32-CAM board to the computer with TYPE-C cable.

⑨ Select your board: Tools > Board > esp32 > ESP32 Dev Module.

⑩Select the Port： Tools >Port > COMx(The COM port number of each computer may be different, and the port number that appears is your port number) (If you do not see COM ports other than COM1 in Arduino IDE, you need to refer to the installation of CH340 driver in "2_Programming_Preparation" and install it.)

⑪ Unplug the serial port transmission cable between the Arduino control board and the ESP32-CAM control board. (This line can be returned after the upload of this program.)

⑫Upload the program to the ESP32-CAM controller board.

The picture above shows that it is uploaded successfully.

⑬After uploading the code successfully. Disconnect the data line, put the car on the ground, and load 2 18650 batteries. (If you buy the battery-free version, please prepare two 18650 batteries.) Then turn on the switch.

⑭Then pick up your cell phone or other mobile devices, turn on the WiFi switch, find "KUONGSHUN-AD176", enter the password "12345678" to connect to the hotspot, and if you are reminded that there is no network, please stay connected. (If you turn on the switch, the module lights up but your cell phone can not search "KUONGSHUN-AD176", please check if the wiring is correct and the battery is charged.)

⑮Open the mobile browser, input 192.168.4.1, jump to the following screen.

⑯Interface Introduction

: Enable image receiving and display. When pressed, the image will appear:

: Screenshot and display.

: Disable image reception.

: Hold to control the car to move forward.

Backward : Hold to control the car to translate backwards.

TurnLeft : Hold to control the car to pan to the left.

TurnRight : Hold to control the car to pan to the right.

Leftup : Hold to control the car to pan left and up.

Rightup : Hold to control the car to pan up right.

Leftdown : Hold to control the car to pan left and down.

Rightdown : Hold to control the car to pan down right.

Free Control : Enter Free Control mode.

Obstacle Avoidance : Enter Obstacle Avoidance mode.

Object Following : Enter Object Following mode.

Line Tracing : Enter Line Tracing mode.

Servo :

Adjust the servo pitch angle of the front of the car.

Speed :

Adjust the speed of the car.

L E D :

Switch headway LED lights and adjust brightness.

Now when you start the car again, you don't need to download the program again.You just start in step ⑭.

Program Explanation:

WiFi Name and Password Modification

In this project, you can modify the WiFi name and password by simply changing the values of *ssid and *password. Here, *ssid refers to the WiFi name, and *password refers to the password.

Note: Here, we are modifying the WiFi name and password broadcasted by the car.

5.2_ESP32Cam_WIFI_Web.ino | CameraWebServer.cpp | app_httpd.cpp

```
#include "esp_system.h"
#include "esp_camera.h"
#include <WiFi.h>

const char *ssid = "KUONGSHUN-AD176";
const char *password = "12345678";


String wifi_name;

#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27
#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22

void startCameraServer();
void CameraWebServer_init()
{
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
```

Change the Video Clarity

By selecting the byte length sample value, you can change the image clarity. There are three clarity options available, and you can only choose one. To do so, comment out the other two options. Generally, the higher the image quality, the more likely there will be a delay in the image display.

```
sensor_t *s = esp_camera_sensor_get();
//drop down frame size for higher initial frame rate
//s->set_framesize(s, FRAMESIZE_SXGA); //Video sample rate:60000      #9 (High image quality)  1280x1024
//s->set_framesize(s, FRAMESIZE_SVGA); //Video sample rate:40000      #7 (Medium image quality)  800x600
s->set_framesize(s, FRAMESIZE_QVGA); //Video sample rate:10000       #4 (Low image quality)  320x240
```

5.2_ESP32Cam_WIFI_Web.ino | CameraWebServer.cpp | app_httpd.cpp

```
69        config.fb_location = CAMERA_FB_IN_DRAM;
70      }
71    } else {
72      // Best option for face detection/recognition
73      config.frame_size = FRAMESIZE_240X240;
74  #if CONFIG_IDF_TARGET_ESP32S3
75      config.fb_count = 2;
76  #endif
77    }
78    // camera init
79    esp_err_t err = esp_camera_init(&config);
80    if (err != ESP_OK)
81    {
82      Serial.printf("Camera init failed with error 0x%x", err);
83      return;
84    }
85    sensor_t *s = esp_camera_sensor_get();
86    //drop down frame size for higher initial frame rate
87    //s->set_framesize(s, FRAMESIZE_SXGA); //Video sample rate:60000      #9 (High image quality)  1280x1024
88    //s->set_framesize(s, FRAMESIZE_SVGA); //Video sample rate:40000      #7 (Medium image quality)  800x600
89    s->set_framesize(s, FRAMESIZE_QVGA); //Video sample rate:10000      #4 (Low image quality)  320x240
90
91    s->set_vflip(s, 1);   //Image Direction Settings(Up/Down)
92    s->set_hmirror(s, 0); //Image Direction Setting(Left/Right)
93
94    WiFi.setTxPower(WIFI_POWER_19_5dBm);
95    WiFi.mode(WIFI_AP);
96    WiFi.softAP(ssid, password, 5);
97    startCameraServer();
```

Change the Image Orientation

```
91    s->set_vflip(s, 1);   //Image Direction Settings(Up/Down)
92    s->set_hmirror(s, 0); //Image Direction Setting(Left/Right)
```

If the image displayed on the web interface is flipped vertically, simply change the original s->set_vflip(s, 1); to s->set_vflip(s, 0);. After completing this step, the image will be displayed correctly.

Similarly, if the image displayed on the web interface is flipped horizontally, simply change the original s->set_hmirror(s, 0); to s->set_hmirror(s, 1);. After completing this step, the image will be displayed correctly.

Note: The same operation applies in the APP-controlled program.

5.2_ESP32Cam_WIFI_Web.ino | CameraWebServer.cpp | app_httpd.cpp

```
      config.fb_location = CAMERA_FB_IN_DRAM;
    }
  } else {
    // Best option for face detection/recognition
    config.frame_size = FRAMESIZE_240X240;
#if CONFIG_IDF_TARGET_ESP32S3
    config.fb_count = 2;
#endif
  }
  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK)
  {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
  sensor_t *s = esp_camera_sensor_get();
  //drop down frame size for higher initial frame rate
  //s->set_framesize(s, FRAMESIZE_SXGA); //Video sample rate:60000
  //s->set_framesize(s, FRAMESIZE_SVGA); //Video sample rate:40000
  s->set_framesize(s, FRAMESIZE_QVGA); //Video sample rate:10000

  s->set_vflip(s, 1);   //Image Direction Settings(Up/Down)
  s->set_hmirror(s, 0); //Image Direction Setting(Left/Right)

  WiFi.setTxPower(WIFI_POWER_19_5dBm);
  WiFi.mode(WIFI_AP);
  WiFi.softAP(ssid, password, 5);
  startCameraServer();
  // WiFi.softAP(ssid, password);
```

Lesson 6 WIFI Camera Multi-Function Program for APP

In this project, we will use the Arduino board combined with the L293D extension board to realize the functions of walking remote control, obstacle avoidance, patrolling and following of the cart. ESP32-CAM is used to realize WiFi connection and control the cart through serial line. At the same time, ESP32-CAM also transmits the images from the camera at the front of the cart to the web server connected to the hotspot in real time. As long as we connect to this WIFI hotspot, we can use the APP to watch the monitoring in real time, and at the same time, we can control the traveling of the cart through the buttons of the APP, as well as adjusting the mode of operation and the speed of the cart, and so on.

Before presenting the multi-function program of the cart, we first upload the program to the cart. The car has two control boards, and we need to upload two programs to each board. Here are the steps to upload an Arduino board and program:

1.Open the '6.1_Arduino_All_APP.ino' program in 'KUONGSHUN AD176 ESP32 WiFi Camera Mecanum 4WD Smart Robot Car\3_Code_and_Tutorial\6.1_Arduino_All_APP'.

2. Connect the Arduino board to the computer with TYPE-C data cable.

3. Select your control board: Tools > Board > Arduino AVR Board > Arduino UNO.

4. Select the Port: Tools > Port > COMx(The COM port number of each computer may be different, and the port number that appears is your port number) (If you do not see COM ports other than COM1 in Arduino IDE, you need to refer to the installation of CH340 driver in "2_Programming_Preparation" and install it.)

5. Unplug the serial port transmission cable between the Arduino board and the ESP32-CAM board.(This cable will not be connected back during all uploading procedures, until finally all procedures are uploaded and the car is controlled by WIFI.)

6. Upload the program to the UNO controller board.

The picture above shows that it is uploaded successfully.

7. After uploading the code successfully.We also need to upload the program to the **ESP32-CAM board** of the locomotive.

Open the '6.2_ESP32Cam_WIFI_APP.ino' program in 'KUONGSHUN AD176 ESP32 WiFi Camera Mecanum 4WD Smart Robot Car\3_Code_and_Tutorial\6.2_ESP32Cam_WIFI_APP'.

8. Connect the ESP32-CAM board to the computer with TYPE-C cable.

9. Select your board: Tools > Board > esp32 > ESP32 Dev Module.

10. Select the Port: Tools > Port > COMx(The COM port number of each computer may be different, and the port number that appears is your port number) (If you do not see COM ports other than COM1 in Arduino IDE, you need to refer to the installation of CH340 driver in "2_Programming_Preparation" and install it.)

11. Unplug the serial port transmission cable between the Arduino control board and the ESP32-CAM control board. (This line can be returned after the upload of this program.)

12. Upload the program to the ESP32-CAM controller board.

The picture above shows that it is uploaded successfully.

13. After uploading the code successfully. Disconnect the data line, put the car on the ground, and load 2 18650 batteries. (If you buy the battery-free version, please prepare two 18650 batteries.) Then turn on the switch.

14. To Install the Application

For Android:

Open the profile folder, copy the apk file in the “4_Android_Software_Installation_Package” folder to your Android phone or tablet computer , and then install it later. (You can send it to your phone or tablet via communication software and open and install it directly, or save it to your phone's memory and find and open it in file management)

Once the installation is complete, the following icon will appear on your phone or tablet:

For the Apple system:

Enter kuongshun in the APP Store search bar, click to get the download and install, install as shown in the right picture.

15. Then pick up your phone, turn on the WiFi switch, find "**KUONGSHUN-AD176**", enter the password "**12345678**" and connect to the hotspot. If you are reminded that there is no network, please keep connected. (If the switch is turned on and the module lights up but the phone cannot search for "KUONGSHUN", please check whether the wiring is correct and whether the battery is powered.)

16. Open the APP

Set Language

Back to Home

Select ESP32-CAM Mecanum Smart Car

Click the icon in the top-right corner to connect to the ESP32-CAM Mecanum Smart Car.

When the "Connected Successfully" message appears, it indicates that the connection was successful. Once connected, the live camera feed will be displayed, and you can start controlling the car as desired.

Note: If the "Connected Fail" message appears, please check if the Wi-Fi connection has been disconnected. After reconnecting to the Wi-Fi, try clicking the icon again to reconnect.

17. Interface Introduction

① Video Display Area

②Camera Tilt Angle Adjustment Slider

Note: When the slider is moved upward, the servo angle decreases, causing the camera's view to tilt downward. Conversely, when the slider is moved downward, the servo angle increases, raising the camera's view upward.

③Video Image Control Panel

: The pause button freezes the video screen.

: The play button opens the video screen.

: The close button closes the video screen.

④ Motion Mode Switch

: Enter free control mode.

: Enter obstacle avoidance mode.

: Enter obstacle following mode.

: Enter line tracing mode.

⑤Direction Control Pad

: Press the arrow keys or drag the slider on the control pad to the corresponding arrow position to move the car in the specified direction.

Note: When controlling the car's movement, it will automatically switch from other modes to free control mode.

: Control the car to turn left in place.

: Control the car to turn right in place.

⑥Headlight Brightness Adjustment slider.

: Slide to adjust headlight brightness.

⑦Speed Control Slider.

: Slide to adjust the speed of the cart.

: Displays the connection status, if WiFi is connected but not connected to the car, you can tap here to connect.

Now when you start the car again, you don't need to download the program again.You just start in step 15.

Frequently Asked Questions

Q: WIFI signal cannot be searched ?

A: This issue may be caused by insufficient power supply. Please check the following:

1. Is the battery properly connected?
2. Does the motor driver board light up normally?
3. Is the wiring correct?

Q: The screen image is inverted ?

A: If your camera module is "ESP32-CAM-V0.3", in order to display the image correctly, you need to modify the instruction in the **CameraWebServer.cpp** file. Change the command that sets the image flip direction from the original:

s->set_vflip(s, 1); to : s->set_vflip(s, 0);

This change should be applied in both the Web version and the App version.

ESP32-CAM-V0.3

6.2_ESP32Cam_WIFI_APP - CameraWebServer.cpp | Arduino IDE 2.3.2

File Edit Sketch Tools Help

Select Board

6.2_ESP32Cam_WIFI_APP.ino | CameraWebServer.cpp | app_httpd.cpp | camera_index.h

```
      config.frame_size = FRAMESIZE_SVGA;
      config.fb_location = CAMERA_FB_IN_DRAM;
    }
  } else {
    // Best option for face detection/recognition
    config.frame_size = FRAMESIZE_240X240;
#if CONFIG_IDF_TARGET_ESP32S3
    config.fb_count = 2;
#endif
  }
  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK)
  {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
  sensor_t *s = esp_camera_sensor_get();
  //drop down frame size for higher initial frame rate
  //s->set_framesize(s, FRAMESIZE_SXGA); //Video sample rate:60000          #9 (High image quality)  1280x1024
  s->set_framesize(s, FRAMESIZE_SVGA); //Video sample rate:40000           #7 (Medium image quality)  800x600
  //s->set_framesize(s, FRAMESIZE_QVGA); //Video sample rate:10000          #4 (Low image quality)  320x240

  s->set_vflip(s, 0);  //Image Direction Settings (Up/Down)
  s->set_hmirror(s, 0); //Image Direction Setting (Left/Right)

  // WiFi.setTxPower(WIFI_POWER_19_5dBm);
  WiFi.mode(WIFI_AP);
  WiFi.softAP(ssid, password, 5);
  // WiFi.softAP(ssid, password);
  Serial.print("\r\n");
  startCameraServer();
  Serial.print("\r\n");
  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.softAPIP());
  Serial.println("' to connect");
}
```

ESP32-CAM-V2.0

6.2_ESP32Cam_WIFI_APP - CameraWebServer.cpp | Arduino IDE 2.3.2

File Edit Sketch Tools Help

Select Board

6.2_ESP32Cam_WIFI_APP.ino | CameraWebServer.cpp | app_httpd.cpp | camera_index.h

```
      config.frame_size = FRAMESIZE_SVGA;
      config.fb_location = CAMERA_FB_IN_DRAM;
    }
  } else {
    // Best option for face detection/recognition
    config.frame_size = FRAMESIZE_240X240;
#if CONFIG_IDF_TARGET_ESP32S3
    config.fb_count = 2;
#endif
  }
  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK)
  {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
  sensor_t *s = esp_camera_sensor_get();
  //drop down frame size for higher initial frame rate
  //s->set_framesize(s, FRAMESIZE_SXGA); //Video sample rate:60000          #9 (High image quality)  1280x1024
  s->set_framesize(s, FRAMESIZE_SVGA); //Video sample rate:40000           #7 (Medium image quality)  800x600
  //s->set_framesize(s, FRAMESIZE_QVGA); //Video sample rate:10000          #4 (Low image quality)  320x240

  s->set_vflip(s, 1);  //Image Direction Settings (Up/Down)
  s->set_hmirror(s, 0); //Image Direction Setting (Left/Right)

  // WiFi.setTxPower(WIFI_POWER_19_5dBm);
  WiFi.mode(WIFI_AP);
  WiFi.softAP(ssid, password, 5);
  // WiFi.softAP(ssid, password);
  Serial.print("\r\n");
  startCameraServer();
  Serial.print("\r\n");
  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.softAPIP());
  Serial.println("' to connect");
}
```

Q: How to adjust the image clarity ?

A: You can change the image clarity by selecting the byte length sample value. There are three levels of clarity to choose from, and you can select only one. To apply the selected clarity, simply comment out the other two lines. Generally, the higher the image quality, the more likely there will be a delay in the image display.

```
//drop down frame size for higher initial frame rate
//s->set_framesize(s, FRAMESIZE_SXGA); //Video sample rate:60000          #9 (High image quality)  1280x1024
s->set_framesize(s, FRAMESIZE_SVGA); //Video sample rate:40000           #7 (Medium image quality)  800x600
//s->set_framesize(s, FRAMESIZE_QVGA); //Video sample rate:10000          #4 (Low image quality)  320x240
```