Car patrol based on three-way patrol module

1. Introduction

Please read the "Motor Introduction and Usage" in the four-way motor driver board document first to understand the motor parameters, wiring method, and power supply voltage you are currently using. This will avoid burning the motherboard or motor.

2. Experimental preparation

Wheeled car chassis V1 four-wheel drive version, 4*310 motors, 7.4V lithium battery, three-way patrol module, MSPM0 robot expansion board (optional), MSPM0G3507 core board (Yahboom).

The relationship between the 4 motor interfaces and the car is as follows:

M1 -> upper left motor (left front wheel of the car)

M2 -> lower left motor (left rear wheel of the car)

M3 -> upper right motor (right front wheel of the car)

M4 -> lower right motor (right rear wheel of the car)

Hardware wiring:

Wiring using MSPM0 robot expansion board

Note: The wire used for the three-way patrol module is: XH2.54 cable (5pin double-headed-all black-reverse 25cm), and the direction of the reverse cable holder is as shown in the following figure

Note: The wire used to connect the MSPM0 robot expansion board to the four-way motor drive module is: XPH2.0-4pin cable, double-headed all black, reverse (200mm), and the direction of the reverse cable holder is as shown in the following figure
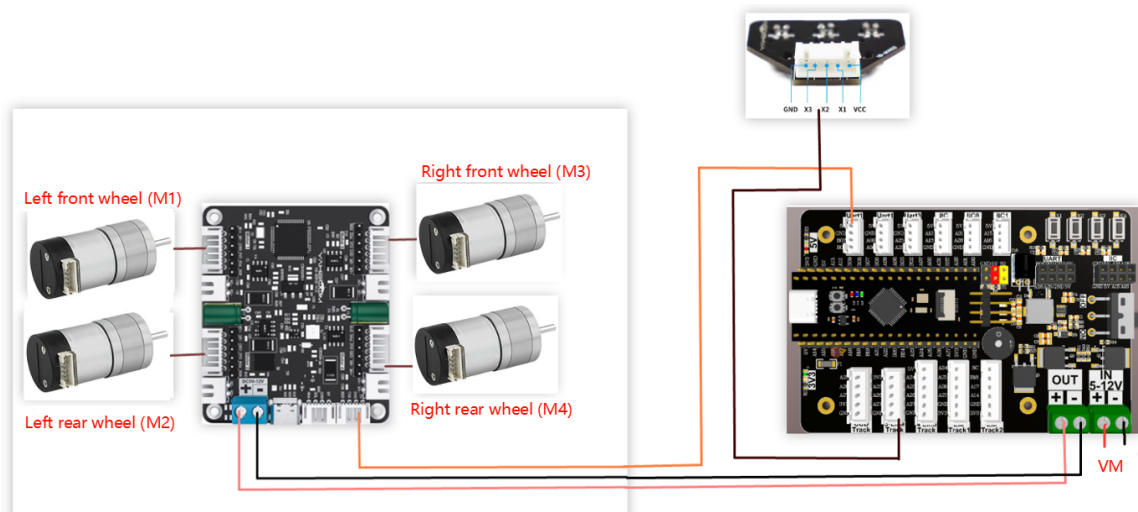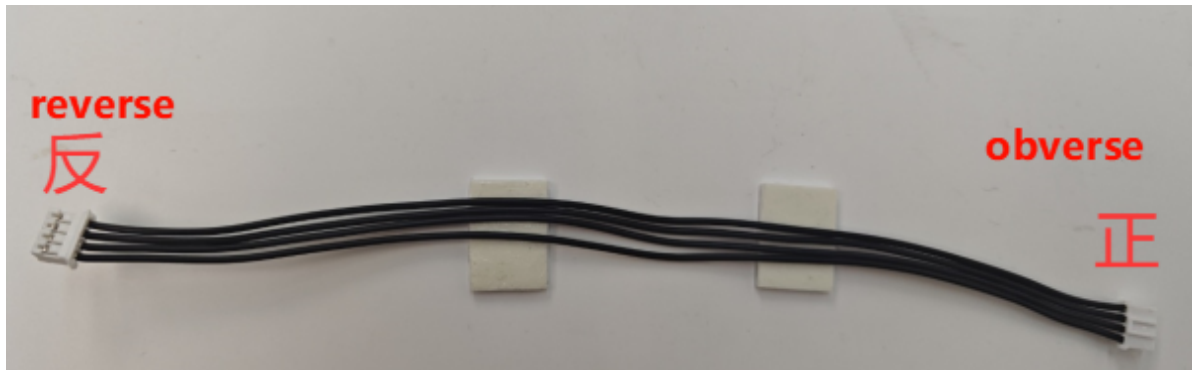
Wiring using MSPM0G3507 core board (Yahboom)
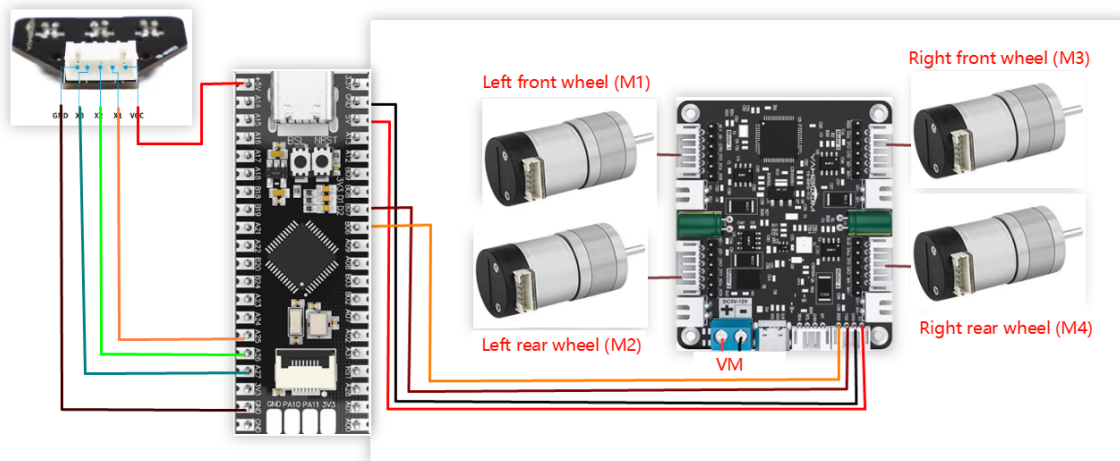
Wiring pins

Four-way motor driver board	MSPM0G3507 core board (Yahboom)
RX2	PB6
TX2	PB7
GND	GND
5V	5V

Take M1 motor as an example below, and other motors are similar

Motor	Four-way motor driver board (Motor)
M+	M1+
M-	M1-
GND	GND
VCC	3V3
B	H1A
A	H1B

Three-way line patrol module	MSPM0G3507 core board (Yahboom)
VCC	5V
X1	PA25
X2	PA26
X3	PA27
GND	GND

3. Key code analysis

- three_linewalking.c

```
//获取X1X2X3的引脚电平 Get the pin levels of X1X2X3
void three_GetLineWalking(int *p_iL, int *p_iM, int *p_iR)
{
    *p_iL = LineWalk_L_IN;
    *p_iM = LineWalk_M_IN;
    *p_iR = LineWalk_R_IN;
}

void three_LineWalking(void)
{
    int LineL = 0, LineM = 0, LineR = 0;

    three_GetLineWalking(&LineL, &LineM, &LineR);//获取黑线检测状态 Get black line
detection status
```

```c
    if( LineL == HIGH &&  LineM == HIGH) //直角锐角   Right angle acute angle
    {
        Contrl_Speed(500,500,500,500); //继续直行   Continue straight

    }
    else if ( LineM == HIGH && LineR == HIGH) //直角锐角    Right angle acute
angle
    {
        Contrl_Speed(500,500,500,500); //继续直行   Continue straight

    }
    else if (LineL == HIGH ) //左最外侧检测微调车左转  Left outermost detection
fine-tuning vehicle turns left
    {
        Contrl_Speed(-500,-500,700,700); //左急转弯 Sharp left turn
    }
    else if ( LineR == HIGH) //右最外侧检测微调车右转  Right outermost detection
fine-tuning vehicle right turn
    {
        Contrl_Speed(700,700,-500,-500); //右急转弯 Sharp right turn
    }
    else if(LineM == HIGH ) // 黑色，加速前进 Black, Speed Up
    {
        Contrl_Speed(500,500,500,500);
    }

}
```

three_GetLineWalking: Get the status of the three-way patrol module

three_LineWalking: Control the car to adjust its driving status according to the different status of the three-way patrol module

- app_motor_usart.c

```c
//发送电机类型    Transmitter motor type
void send_motor_type(motor_type_t data)
{
    sprintf((char*)send_buff,"$mtype:%d#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));

}

//发送电机死区    Send motor dead zone
void send_motor_deadzone(uint16_t data)
{
    sprintf((char*)send_buff,"$deadzone:%d#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));
}

//发送电机磁环脉冲  Send motor magnetic ring pulse
void send_pulse_line(uint16_t data)
{
    sprintf((char*)send_buff,"$mline:%d#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));
}
```

```c
//发送电机减速比    Transmitting motor reduction ratio
void send_pulse_phase(uint16_t data)
{
    sprintf((char*)send_buff,"$mphase:%d#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));
}

//发送轮子直径     Send wheel diameter
void send_wheel_diameter(float data)
{
    sprintf((char*)send_buff,"$wdiameter:%.3f#",data);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));
}

...

/控制速度    Controlling Speed
void Contrl_Speed(int16_t M1_speed,int16_t M2_speed,int16_t M3_speed,int16_t
M4_speed)
{

sprintf((char*)send_buff,"$spd:%d,%d,%d,%d#",M1_speed,M2_speed,M3_speed,M4_speed
);
    Send_Motor_ArrayU8(send_buff, strlen((char*)send_buff));
}
```

Configure the motor parameters of the 4-way motor driver board

Contrl_Speed: Control the speed of the 4 motors

- empty.c

```c
#define MOTOR_TYPE 2    //1:520电机 2:310电机 3:测速码盘TT电机 4:TT直流减速电机 5:L型
520电机
                        //1:520 motor 2:310 motor 3:speed code disc TT motor 4:TT
DC reduction motor 5:L type 520 motor

int main(void)
{
    USART_Init();

    printf("please wait...");
    ...
    #elif MOTOR_TYPE == 2
    send_motor_type(2);
    delay_ms(100);
    send_pulse_phase(20);
    delay_ms(100);
    send_pulse_line(13);
    delay_ms(100);
    send_wheel_diameter(48.00);
    delay_ms(100);
```

```
    send_motor_deadzone(1600);
    delay_ms(100);
    ...
    #endif

    while(1)
    {
        three_LineWalking();
    }

}
```

MOTOR_TYPE: Motor type, change to different values according to your own motor

USART_Init: Initialize the serial port for communicating with the four-way motor driver board

three_LineWalking: Start line patrol

4. Experimental phenomenon

After connecting the car and burning the program to MSPM0, place the car on the map with white background and black lines, with the sensor in the middle of the line patrol module facing the black line, plug in the power supply, and the car will patrol the black line automatically.