CCD Tracking

1. Opening Notes

Please read the "Introduction to Motors and Usage" in the information of the four-way motor driver board first to understand the motor parameters, wiring methods, and power supply voltage you are currently using. To avoid burning the motherboard or motor.

Motor: The case and code take the 310 motor of our store as an example.

2. Experimental Preparation

National Race Chassis V2 Four-wheel Drive Version, 4*310 Motor, 7.4V Lithium Battery, Linear CCD Camera Module, STM32F103C8T6 Core Board

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

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

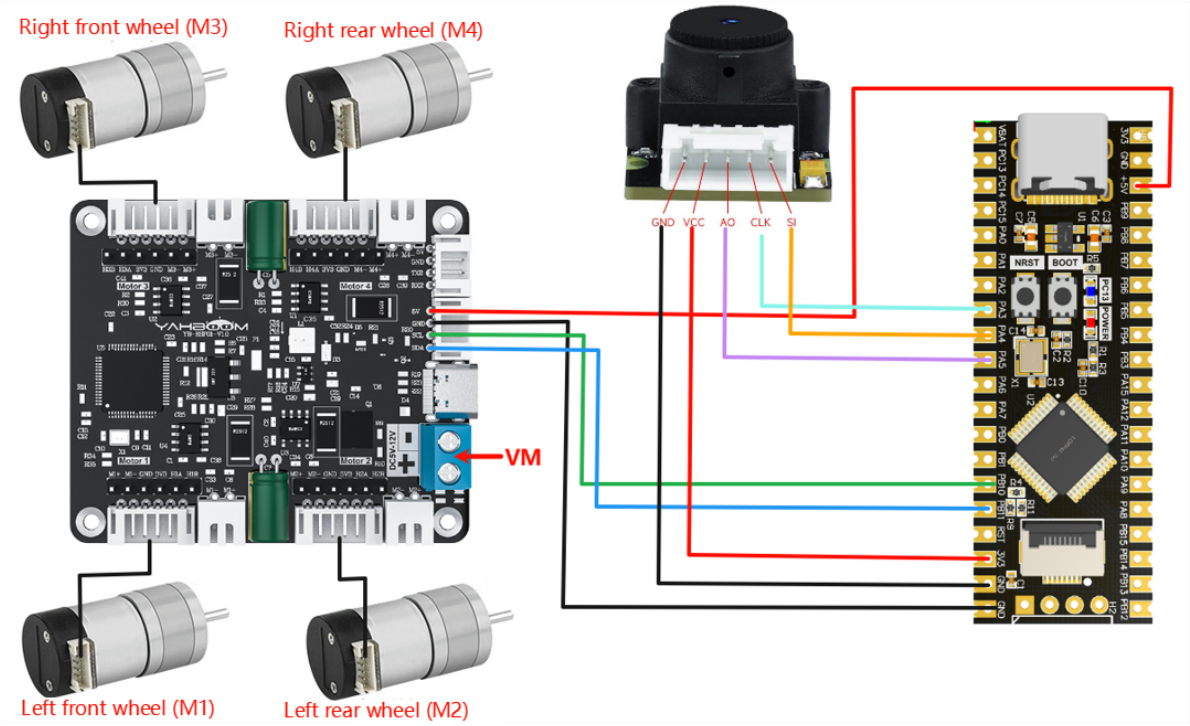
M2 -> lower left motor (left rear wheel of the car)

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

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

Hardware wiring:

Overall wiring



Wiring pins

Four-way motor driver board	STM32C8T6
5V	5V
GND	GND
SCL	PB10
SDA	PB11

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

Motor	Four-way motor driver board (Motor)
M2	M1-
VCC	3V3
A	H1A
B	H1B
GND	GND
M1	M1+

CCD	STM32C8T6
SI	PA4
CLK	PA3
AO	PA4
VCC	3V3
GND	GND

3. Key code analysis

- system.c

```
#include "system.h"
u16 ADV[128]={0};
u8 SciBuf[200];  //存储上传到上位机的信息 Stores information uploaded to the host computer
u8 CCD_Zhongzhi,CCD_Yuzhi;   //线性CCD  相关  Linear CCD  Related
void systemInit(void)
{
    //Delay function initialization
    //延时函数初始化
    delay_init();

    //Interrupt priority group setting
    //中断优先级分组设置
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2);


    delay_ms(500);                  //=====延时等待系统稳定    Delay waiting for system stabilization

    uart_init(115200);          //串口1初始化 Serial port 1 initialization

    ccd_Init();                          //初始化CCD    Initialize CCD
    IIC_Motor_Init();
}
```

System-level initialization, including preparation of modules such as serial port, CCD, I2C motor, delay function, etc.

- adc.c

```
/**************************************************************************
函数功能：线性CCD取中值   Function: Linear CCD median
入口参数：无           Input parameter: None
返回  值：无          Return value: None
**************************************************************************/
void  Find_CCD_Zhongzhi(void)
{
     static uint16_t i,j,Left,Right;
     static uint16_t value1_max,value1_min;
```

```
        value1_max=ADV[0];  //动态阈值算法，读取最大和最小值    Dynamic threshold
algorithm, reads maximum and minimum values
     for(i=5;i<123;i++)   //两边各去掉5个点    Remove 5 points from each side
     {
        if(value1_max<=ADV[i])
        value1_max=ADV[i];
     }
       value1_min=ADV[0];  //最小值    Minimum
     for(i=5;i<123;i++)
     {
        if(value1_min>=ADV[i])
        value1_min=ADV[i];
     }
   CCD_Yuzhi=(value1_max+value1_min)/2;   //计算出本次中线提取的阈值    Calculate the
threshold for this midline extraction
     for(i = 5;i<118; i++)   //寻找左边跳变沿  Find the left edge
    {
        if(ADV[i]>CCD_Yuzhi&&ADV[i+1]>CCD_Yuzhi&&ADV[i+2]>CCD_Yuzhi&&ADV[i+3]
<CCD_Yuzhi&&ADV[i+4]<CCD_Yuzhi&&ADV[i+5]<CCD_Yuzhi)
        {
            Left=i;
            break;
        }
    }
     for(j = 108;j>15; j--)//寻找右边跳变沿    Find the right edge
  {
        if(ADV[j]<CCD_Yuzhi&&ADV[j+1]<CCD_Yuzhi&&ADV[j+2]
<CCD_Yuzhi&&ADV[j+3]>CCD_Yuzhi&&ADV[j+4]>CCD_Yuzhi&&ADV[j+5]>CCD_Yuzhi)
        {
          Right=j;
          break;
        }
  }
    CCD_Zhongzhi=(Right+Left)/2;//计算中线位置    Calculate the centerline position
}

//开始CCD采集并处理输出数据    Start CCD acquisition and process output data
void deal_data_ccd(void)
{
        RD_TSL();
        Find_CCD_Zhongzhi();
}

void use_ccd_line_motion_PID(void)
{
    g_ccd_median=CCD_Zhongzhi - 55;
    pid_output_ele = (int)(APP_ELE_PID_Calc(g_ccd_median));//位置式PID Position
PID
    motion_car_control(CCD_SPEED, 0, pid_output_ele);//直接控制电机       Direct
motor control

}
```

By reading the CCD sensor data to obtain the center offset value, the position PID control algorithm is used to calculate the steering amount in real time, and then the motor movement is adjusted to achieve tracking control.

use_ccd_line_motion_PID: After obtaining the median value, the position PID is used to calculate the speed of each motor to drive the motor. According to the median value obtained by your own CCD in the middle of the track, the value of 55 can be modified to your own value.

app_motor.c

```
#define CCD_PID_KP       (90)
#define CCD_PID_KI       (0)
#define CCD_PID_KD       (5)


float APP_ELE_PID_Calc(int8_t actual_value)
{
    float pid_out = 0;
    int8_t error;
    static int8_t error_last=0;
    static float Integral;//积分    integral

    error=actual_value;

    Integral +=error;

    //位置式pid Positional pid
    pid_out=error*CCD_PID_KP
                            +CCD_PID_KI*Integral
                            +(error - error_last)*CCD_PID_KD;

    return pid_out;
}
```

Here, modify the PID effect of CCD line patrol. If the line patrol effect is not good, you can set KP and KD to 0 first, then slowly increase KP, and finally try to increase the value of KD

- bsp_motor_iic.c

```
//配置电机  Configure the motor
void Set_motor_type(uint8_t data)
{
    i2cWrite(Motor_model_ADDR,MOTOR_TYPE_REG,2,&data);
}

//配置死区  Configuring Dead Zone
void Set_motor_deadzone(uint16_t data)
{
    static uint8_t buf_tempzone[2];

    buf_tempzone[0] = (data>>8)&0xff;
    buf_tempzone[1] = data;

    i2cWrite(Motor_model_ADDR,MOTOR_DeadZONE_REG,2,buf_tempzone);
}

//配置磁环线 Configuring magnetic loop
```

```
void Set_Pluse_line(uint16_t data)
{
    static uint8_t buf_templine[2];

    buf_templine[0] = (data>>8)&0xff;
    buf_templine[1] = data;

    i2cWrite(Motor_model_ADDR,MOTOR_PluseLine_REG,2,buf_templine);
}

//配置减速比 Configure the reduction ratio
void Set_Pluse_Phase(uint16_t data)
{
    static uint8_t buf_tempPhase[2];

    buf_tempPhase[0] = (data>>8)&0xff;
    buf_tempPhase[1] = data;

    i2cWrite(Motor_model_ADDR,MOTOR_PlusePhase_REG,2,buf_tempPhase);
}


//配置直径  Configuration Diameter
void Set_Wheel_dis(float data)
{
    static uint8_t bytes[4];

    float_to_bytes(data,bytes);

    i2cWrite(Motor_model_ADDR,WHEEL_DIA_REG,4,bytes);
}
```

Define the function of writing configuration parameters to the four-way motor driver board to set key parameters such as motor type, dead zone, number of magnetic ring lines, reduction ratio and wheel diameter to ensure that the motor can be controlled and moved accurately as expected.

- main.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)
{
    systemInit();                   //Initialization function     初始化函数

    #if MOTOR_TYPE == 1
    ...

    #elif MOTOR_TYPE == 2
    Set_motor_type(2);//配置电机类型  Configure motor type
    delay_ms(100);
    Set_Pluse_Phase(20);//配置减速比 查电机手册得出 Configure the reduction ratio. Check the motor manual to find out
    delay_ms(100);
```

```
    Set_Pluse_line(13);//配置磁环线 查电机手册得出  Configure the magnetic ring wire.
Check the motor manual to get the result.
    delay_ms(100);
    Set_Wheel_dis(48.00);//配置轮子直径,测量得出      Configure the wheel diameter
and measure it
    delay_ms(100);
    Set_motor_deadzone(1300);//配置电机死区,实验得出  Configure the motor dead zone,
and the experiment shows
    delay_ms(100);

    ...
    #endif

    while(1)
    {
        deal_data_ccd();
        use_ccd_line_motion_PID();
        delay_ms(6);
    }
}
```

MOTOR_TYPE: used to set the type of motor used. Modify the corresponding number according to the comments based on the motor you are currently using.

Initialize the system through the `systemInit()` function, then configure different types of motors according to the preset `MOTOR_TYPE`, and set the relevant parameters of the motor, such as the reduction ratio, magnetic ring line, wheel diameter, and motor dead zone. In the main loop, call `deal_data_ccd()` regularly to obtain CCD acquisition data and calculate the centerline position of the track, then adjust the direction of the car through the PID algorithm, and finally control the motor through `motion_car_control()` to make adjustments to ensure that the car drives along the track.

4. Experimental phenomenon

After connecting the car and burning the program to the STM32, put the car on the map with a white background and black lines, aim the camera at the black line, connect the power supply, and the car will patrol the black line and drive automatically.