# 目录

1. 控制小车行动综合代码
2. WiFi控制代码
3. 寻迹代码
4. 超声波舵机云台代码

# 控制小车行动综合代码

#include <LiquidCrystal.h>

LiquidCrystal lcd(13,12,7,6,5,4,3);

int Echo = A1; // Echo回声脚(P2.0)

int Trig =A0; // Trig 触发脚(P2.1)

int chaoshengboFront\_Distance = 0;

int chaoshengboLeft\_Distance = 0;

int chaoshengboRight\_Distance = 0;

int Left\_motor=8; //左电机(IN3) 输出0 前进 输出1 后退

int Left\_motor\_pwm=9; //左电机PWM调速

int Right\_motor\_pwm=10; // 右电机PWM调速

int Right\_motor=11; // 右电机后退(IN1) 输出0 前进 输出1 后退

int key=A2;//定义按键 数字A2 接口

int beep=A3;//定义蜂鸣器 数字A3 接口

const int SensorRight = 3; //右循迹红外传感器(P3.2 OUT1)

const int SensorLeft = 4; //左循迹红外传感器(P3.3 OUT2)

int SL; //左循迹红外传感器状态

int SR; //右循迹红外传感器状态

int chaoshengboservopin=2;//设置舵机驱动脚到数字口2

int servopin7=7;//设置左右舵机驱动脚到数字口7

int servopin12=12;//设置上下舵机驱动脚到数字口12

int myangle;//定义角度变量

int pulsewidth;//定义脉宽变量

int val;

char buffer[18]; //串口缓冲区的字符数组

void setup()

{

Serial.begin(9600); // 初始化串口

//初始化电机驱动IO为输出方式

pinMode(Left\_motor,OUTPUT); // PIN 8 8脚无PWM功能

pinMode(Left\_motor\_pwm,OUTPUT); // PIN 9 (PWM)

pinMode(Right\_motor\_pwm,OUTPUT);// PIN 10 (PWM)

pinMode(Right\_motor,OUTPUT);// PIN 11 (PWM)

pinMode(key,INPUT);//定义按键接口为输入接口

pinMode(beep,OUTPUT);

pinMode(SensorRight, INPUT); //定义右循迹红外传感器为输入

pinMode(SensorLeft, INPUT); //定义左循迹红外传感器为输入

//初始化超声波引脚

pinMode(Echo, INPUT); // 定义超声波输入脚

pinMode(Trig, OUTPUT); // 定义超声波输出脚

lcd.begin(16,2); //初始化1602液晶工作 模式

//定义1602液晶显示范围为2行16列字符

pinMode(chaoshengboservopin,OUTPUT);//设定舵机接口为输出接口

pinMode(servopin7,OUTPUT);//设定舵机接口为输出接口

pinMode(servopin12,OUTPUT);//设定舵机接口为输出接口

}

void run() // 前进

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void brake() //刹车，停车

{

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM 调速输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM 调速输出0

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100);//执行时间，可以调整

}

void left() //左转(左轮不动，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void spin\_left() //左转(左轮后退，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,50);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void right() //右转(右轮不动，左轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机不转

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void spin\_right() //右转(右轮后退，左轮前进)

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机PWM输出1

analogWrite(Right\_motor\_pwm,50);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void back() //后退

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void chaoshengbobrake(int time) //刹车，停车

{

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM 调速输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM 调速输出0

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100);//执行时间，可以调整

}

void chaoshengbospin\_left(int time) //左转(左轮后退，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100); //执行时间，可以调整

}

void chaoshengbospin\_right(int time) //右转(右轮后退，左轮前进)

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机PWM输出1

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100); //执行时间，可以调整

}

void chaoshengboback(int time) //后退

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100); //执行时间，可以调整

}

void keysacn()//按键扫描

{

int val;

val=digitalRead(key);//读取数字7 口电平值赋给val

while(!digitalRead(key))//当按键没被按下时，一直循环

{

val=digitalRead(key);//此句可省略，可让循环跑空

}

while(digitalRead(key))//当按键被按下时

{

delay(10); //延时10ms

val=digitalRead(key);//读取数字7 口电平值赋给val

if(val==HIGH) //第二次判断按键是否被按下

{

digitalWrite(beep,HIGH); //蜂鸣器响

while(!digitalRead(key)) //判断按键是否被松开

digitalWrite(beep,LOW); //蜂鸣器停止

}

else

digitalWrite(beep,LOW); //蜂鸣器停止

}

}

float Distance\_test() // 量出前方距离

{

digitalWrite(Trig, LOW); // 给触发脚低电平2μs

delayMicroseconds(2);

digitalWrite(Trig, HIGH); // 给触发脚高电平10μs，这里至少是10μs

delayMicroseconds(10);

digitalWrite(Trig, LOW); // 持续给触发脚低电

float Fdistance = pulseIn(Echo, HIGH); // 读取高电平时间(单位：微秒)

Fdistance= Fdistance/58; //为什么除以58等于厘米， Y米=（X秒\*344）/2

// X秒=（ 2\*Y米）/344 ==》X秒=0.0058\*Y米 ==》厘米=微秒/58

Serial.print("Distance:"); //输出距离（单位：厘米）

Serial.println(Fdistance); //显示距离

//Distance = Fdistance;

return Fdistance;

}

void Distance\_display(int Distance)//显示距离

{

if((2<Distance)&(Distance<400))

{

lcd.home(); //把光标移回左上角，即从头开始输出

lcd.print(" Distance: "); //显示

lcd.setCursor(6,2); //把光标定位在第2行，第6列

lcd.print(Distance); //显示距离

lcd.print("cm"); //显示

}

else

{

lcd.home(); //把光标移回左上角，即从头开始输出

lcd.print("!!! Out of range"); //显示

}

delay(250);

lcd.clear();

}

void servopulse(int chaoshengboservopin,int myangle)/\*定义一个脉冲函数，用来模拟方式产生PWM值舵机的范围是0.5MS到2.5MS 1.5MS 占空比是居中周期是20MS\*/

{

pulsewidth=(myangle\*11)+500;//将角度转化为500-2480 的脉宽值 这里的myangle就是0-180度 所以180\*11+50=2480 11是为了换成90度的时候基本就是1.5MS

digitalWrite(chaoshengboservopin,HIGH);//将舵机接口电平置高 90\*11+50=1490uS 就是1.5ms

delayMicroseconds(pulsewidth);//延时脉宽值的微秒数 这里调用的是微秒延时函数

digitalWrite(chaoshengboservopin,LOW);//将舵机接口电平置低

// delay(20-pulsewidth/1000);//延时周期内剩余时间 这里调用的是ms延时函数

delay(20-(pulsewidth\*0.001));//延时周期内剩余时间 这里调用的是ms延时函数

}

void chaoshengbofront\_detection()

{

//此处循环次数减少，为了增加小车遇到障碍物的反应速度

for(int i=0;i<=5;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulse(chaoshengboservopin,90);//模拟产生PWM

}

chaoshengboFront\_Distance = Distance\_test();

Serial.print("chaoshengboFront\_Distance:"); //输出距离（单位：厘米）

Serial.println(chaoshengboFront\_Distance); //显示距离

//Distance\_display(Front\_Distance);

}

void chaoshengboleft\_detection()

{

for(int i=0;i<=15;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulse(chaoshengboservopin,175);//模拟产生PWM

}

chaoshengboLeft\_Distance = Distance\_test();

Serial.print("chaoshengboLeft\_Distance:"); //输出距离（单位：厘米）

Serial.println(chaoshengboLeft\_Distance); //显示距离

}

void chaoshengboright\_detection()

{

for(int i=0;i<=15;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulse(chaoshengboservopin,5);//模拟产生PWM

}

chaoshengboRight\_Distance = Distance\_test();

Serial.print("chaoshengboRight\_Distance:"); //输出距离（单位：厘米）

Serial.println(chaoshengboRight\_Distance); //显示距离

}

void WIFIservopulse(int servopin7,int myangle)/\*定义一个脉冲函数，用来模拟方式产生PWM值舵机的范围是0.5MS到2.5MS 1.5MS 占空比是居中周期是20MS\*/

{

pulsewidth=(myangle\*11)+450;//将角度转化为500-2480 的脉宽值 这里的myangle就是0-180度 所以180\*11+50=2480 11是为了换成90度的时候基本就是1.5MS

digitalWrite(servopin7,HIGH);//将舵机接口电平置高 90\*11+50=1490uS 就是1.5ms

delayMicroseconds(pulsewidth);//延时脉宽值的微秒数 这里调用的是微秒延时函数

digitalWrite(servopin7,LOW);//将舵机接口电平置低

// delay(20-pulsewidth/1000);//延时周期内剩余时间 这里调用的是ms延时函数

delay(20-(pulsewidth\*0.001));//延时周期内剩余时间 这里调用的是ms延时函数

}

void WIFIservopulsesx(int servopin12,int myangle)/\*定义一个脉冲函数，用来模拟方式产生PWM值舵机的范围是0.5MS到2.5MS 1.5MS 占空比是居中周期是20MS\*/

{

pulsewidth=(myangle\*11)+400;//将角度转化为500-2480 的脉宽值 这里的myangle就是0-180度 所以180\*11+50=2480 11是为了换成90度的时候基本就是1.5MS

digitalWrite(servopin12,HIGH);//将舵机接口电平置高 90\*11+50=1490uS 就是1.5ms

delayMicroseconds(pulsewidth);//延时脉宽值的微秒数 这里调用的是微秒延时函数

digitalWrite(servopin12,LOW);//将舵机接口电平置低

// delay(20-pulsewidth/1000);//延时周期内剩余时间 这里调用的是ms延时函数

delay(20-(pulsewidth\*0.001));//延时周期内剩余时间 这里调用的是ms延时函数

}

void WIFIfront\_detection()// 左右电机前

{

//此处循环次数减少，为了增加小车遇到障碍物的反应速度

for(int i=0;i<=5;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

WIFIservopulse(servopin7,90);//模拟产生PWM

}

}

void WIFIleft\_detection()//左右舵机靠左

{

for(int i=0;i<=1;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

WIFIservopulse(servopin7,175);//模拟产生PWM

}

}

void WIFIright\_detection()//左右电机靠右

{

for(int i=0;i<=1;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

WIFIservopulse(servopin7,1);//模拟产生PWM

}

}

void WIFIs\_detection()//上下舵机上

{

for(int i=0;i<=1;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

WIFIservopulsesx(servopin12,0);//模拟产生PWM

}

}

void WIFIx\_detection()//上下舵机下

{

for(int i=0;i<=1;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

WIFIservopulsesx(servopin12,175);//模拟产生PWM

}

}

void loop()

{

if(Serial.available() > 0) //Serial.available()返回串口收到的字节数

{

int index = 0;

delay(100); //延时等待串口收完数据，否则刚收到1个字节时就会执行后续程序

int numChar = Serial.available();

if(numChar > 15) //确认数据不会溢出，应当小于缓冲大小

{

numChar = 15;

}

while(numChar--)

{

buffer[index++] = Serial.read(); //将串口数据一字一字的存入缓冲

}

splitString(buffer); //字符串分割

}

keysacn(); //调用按键扫描函数

while(1)

{

SR = digitalRead(SensorRight);//有信号表明在白色区域，车子底板上L1亮；没信号表明压在黑线上，车子底板上L1灭

SL = digitalRead(SensorLeft);//有信号表明在白色区域，车子底板上L2亮；没信号表明压在黑线上，车子底板上L2灭

chaoshengbofront\_detection();//测量前方距离

if(chaoshengboFront\_Distance < 30)//当遇到障碍物时

{

chaoshengbobrake(2);//先刹车

chaoshengboback(2);//后退减速

chaoshengbobrake(2);//停下来做测距

chaoshengboleft\_detection();//测量左边距障碍物距离

Distance\_display(chaoshengboLeft\_Distance);//液晶屏显示距离

chaoshengboright\_detection();//测量右边距障碍物距离

Distance\_display(chaoshengboRight\_Distance);//液晶屏显示距离

if((chaoshengboLeft\_Distance < 30 ) &&( chaoshengboRight\_Distance < 30 ))//当左右两侧均有障碍物靠得比较近

chaoshengbospin\_left(18);//旋转掉头

else if(chaoshengboLeft\_Distance > chaoshengboRight\_Distance)//左边比右边空旷

{

chaoshengbospin\_left(10);//左转

chaoshengbobrake(1);//刹车，稳定方向

}

else//右边比左边空旷

{

chaoshengbospin\_right(10);//右转

chaoshengbobrake(1);//刹车，稳定方向

}

}

else

{

run(); //无障碍物，直行

}

if (SL == LOW&&SR==LOW)

run(); //调用前进函数

else if (SL == HIGH & SR == LOW)// 左循迹红外传感器,检测到信号，车子向右偏离轨道，向左转

spin\_left();

else if (SR == HIGH & SL == LOW) // 右循迹红外传感器,检测到信号，车子向左偏离轨道，向右转

spin\_right();

else // 都是黑色, 停止

brake();

}

}

void splitString(char \*data)

{

Serial.print("Data entered:");

Serial.println(data);

char \*parameter;

parameter = strtok(data, " ,"); //string token，将data按照空格或者,进行分割并截取

Serial.print("\*\*\*");

Serial.println(parameter);

while(parameter != NULL)

{

setLED(parameter);

parameter = strtok(NULL, " ,"); //string token，再次分割并截取，直至截取后的字符为空

Serial.print("---");

Serial.println(parameter);

}

for(int x = 0; x < 16; x++) //清空缓冲

{

buffer[x] = '\0';

}

Serial.flush();

}

void setLED(char \*data)

{

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'A'))

{

Serial.println("go forward!");

run();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'B'))

{

Serial.println("go back!");

back();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'C'))

{

Serial.println("go left!");

left();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'D'))

{

Serial.println("go right!");

right();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'E'))

{

Serial.println("Stop!");

brake();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'F'))

{

Serial.println("Stop!");

brake();

}

/\* 以下是控制舵机左，上下舵机 \*/

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'L'))//左

{

WIFIleft\_detection();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'I'))//右

{

WIFIright\_detection();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'J'))//上

{

WIFIs\_detection();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'K'))//下

{

WIFIx\_detection();

}

}

# WiFi视频传输控制代码

int Left\_motor=8; //左电机(IN3) 输出0 前进 输出1 后退

int Left\_motor\_pwm=9; //左电机PWM调速

int Right\_motor\_pwm=10; // 右电机PWM调速

int Right\_motor=11; // 右电机后退(IN1) 输出0 前进 输出1 后退

int servopin7=7;//设置左右舵机驱动脚到数字口7

int servopin12=12;//设置上下舵机驱动脚到数字口12

int myangle;//定义角度变量

int pulsewidth;//定义脉宽变量

int val;

char buffer[18]; //串口缓冲区的字符数组

void setup() //设定串口和引脚模式

{

Serial.begin(9600);

Serial.flush(); //清空串口缓存

pinMode(Left\_motor,OUTPUT); // PIN 8 8脚无PWM功能

pinMode(Left\_motor\_pwm,OUTPUT); // PIN 9 (PWM)

pinMode(Right\_motor\_pwm,OUTPUT);// PIN 10 (PWM)

pinMode(Right\_motor,OUTPUT);// PIN 11 (PWM)

pinMode(servopin7,OUTPUT);//设定舵机接口为输出接口

pinMode(servopin12,OUTPUT);//设定舵机接口为输出接口

}

void run() // 前进

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void brake() //刹车，停车

{

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM 调速输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM 调速输出0

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100);//执行时间，可以调整

}

void left() //左转(左轮不动，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void spin\_left() //左转(左轮后退，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void right() //右转(右轮不动，左轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机不转

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void spin\_right() //右转(右轮后退，左轮前进)

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机PWM输出1

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void back() //后退

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void servopulse(int servopin7,int myangle)/\*定义一个脉冲函数，用来模拟方式产生PWM值舵机的范围是0.5MS到2.5MS 1.5MS 占空比是居中周期是20MS\*/

{

pulsewidth=(myangle\*11)+450;//将角度转化为500-2480 的脉宽值 这里的myangle就是0-180度 所以180\*11+50=2480 11是为了换成90度的时候基本就是1.5MS

digitalWrite(servopin7,HIGH);//将舵机接口电平置高 90\*11+50=1490uS 就是1.5ms

delayMicroseconds(pulsewidth);//延时脉宽值的微秒数 这里调用的是微秒延时函数

digitalWrite(servopin7,LOW);//将舵机接口电平置低

// delay(20-pulsewidth/1000);//延时周期内剩余时间 这里调用的是ms延时函数

delay(20-(pulsewidth\*0.001));//延时周期内剩余时间 这里调用的是ms延时函数

}

void servopulsesx(int servopin12,int myangle)/\*定义一个脉冲函数，用来模拟方式产生PWM值舵机的范围是0.5MS到2.5MS 1.5MS 占空比是居中周期是20MS\*/

{

pulsewidth=(myangle\*11)+400;//将角度转化为500-2480 的脉宽值 这里的myangle就是0-180度 所以180\*11+50=2480 11是为了换成90度的时候基本就是1.5MS

digitalWrite(servopin12,HIGH);//将舵机接口电平置高 90\*11+50=1490uS 就是1.5ms

delayMicroseconds(pulsewidth);//延时脉宽值的微秒数 这里调用的是微秒延时函数

digitalWrite(servopin12,LOW);//将舵机接口电平置低

// delay(20-pulsewidth/1000);//延时周期内剩余时间 这里调用的是ms延时函数

delay(20-(pulsewidth\*0.001));//延时周期内剩余时间 这里调用的是ms延时函数

}

void front\_detection()// 左右电机前

{

//此处循环次数减少，为了增加小车遇到障碍物的反应速度

for(int i=0;i<=5;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulse(servopin7,90);//模拟产生PWM

}

}

void left\_detection()//左右舵机靠左

{

for(int i=0;i<=1;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulse(servopin7,175);//模拟产生PWM

}

}

void right\_detection()//左右电机靠右

{

for(int i=0;i<=1;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulse(servopin7,1);//模拟产生PWM

}

}

void s\_detection()//上下舵机上

{

for(int i=0;i<=1;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulsesx(servopin12,0);//模拟产生PWM

}

}

void x\_detection()//上下舵机下

{

for(int i=0;i<=1;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulsesx(servopin12,175);//模拟产生PWM

}

}

void loop()

{

if(Serial.available() > 0) //Serial.available()返回串口收到的字节数

{

int index = 0;

delay(100); //延时等待串口收完数据，否则刚收到1个字节时就会执行后续程序

int numChar = Serial.available();

if(numChar > 15) //确认数据不会溢出，应当小于缓冲大小

{

numChar = 15;

}

while(numChar--)

{

buffer[index++] = Serial.read(); //将串口数据一字一字的存入缓冲

}

splitString(buffer); //字符串分割

}

}

void splitString(char \*data)

{

Serial.print("Data entered:");

Serial.println(data);

char \*parameter;

parameter = strtok(data, " ,"); //string token，将data按照空格或者,进行分割并截取

Serial.print("\*\*\*");

Serial.println(parameter);

while(parameter != NULL)

{

setLED(parameter);

parameter = strtok(NULL, " ,"); //string token，再次分割并截取，直至截取后的字符为空

Serial.print("---");

Serial.println(parameter);

}

for(int x = 0; x < 16; x++) //清空缓冲

{

buffer[x] = '\0';

}

Serial.flush();

}

void setLED(char \*data)

{

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'A'))

{

Serial.println("go forward!");

run();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'B'))

{

Serial.println("go back!");

back();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'C'))

{

Serial.println("go left!");

left();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'D'))

{

Serial.println("go right!");

right();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'E'))

{

Serial.println("Stop!");

brake();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'F'))

{

Serial.println("Stop!");

brake();

}

/\* 以下是控制舵机左，上下舵机 \*/

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'L'))//左

{

left\_detection();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'I'))//右

{

right\_detection();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'J'))//上

{

s\_detection();

}

if((data[0] == 'O') && (data[1] == 'N')&& (data[2] == 'K'))//下

{

x\_detection();

}

}

# 寻迹控制代码

int Left\_motor=8; //左电机(IN3) 输出0 前进 输出1 后退

int Left\_motor\_pwm=9; //左电机PWM调速

int Right\_motor\_pwm=10; // 右电机PWM调速

int Right\_motor=11; // 右电机后退(IN1) 输出0 前进 输出1 后退

int key=A2;//定义按键 数字A2 接口

int beep=A3;//定义蜂鸣器 数字A3 接口

const int SensorRight = 3; //右循迹红外传感器(P3.2 OUT1)

const int SensorLeft = 4; //左循迹红外传感器(P3.3 OUT2)

int SL; //左循迹红外传感器状态

int SR; //右循迹红外传感器状态

void setup()

{

//初始化电机驱动IO为输出方式

pinMode(Left\_motor,OUTPUT); // PIN 8 8脚无PWM功能

pinMode(Left\_motor\_pwm,OUTPUT); // PIN 9 (PWM)

pinMode(Right\_motor\_pwm,OUTPUT);// PIN 10 (PWM)

pinMode(Right\_motor,OUTPUT);// PIN 11 (PWM)

pinMode(key,INPUT);//定义按键接口为输入接口

pinMode(beep,OUTPUT);

pinMode(SensorRight, INPUT); //定义右循迹红外传感器为输入

pinMode(SensorLeft, INPUT); //定义左循迹红外传感器为输入

}

//=======================智能小车的基本动作=========================

//void run(int time) // 前进

void run() // 前进

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void brake() //刹车，停车

{

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM 调速输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM 调速输出0

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100);//执行时间，可以调整

}

void left() //左转(左轮不动，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void spin\_left() //左转(左轮后退，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,180);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,80);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void right() //右转(右轮不动，左轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机不转

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void spin\_right() //右转(右轮后退，左轮前进)

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机PWM输出1

analogWrite(Right\_motor\_pwm,80);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,180);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void back() //后退

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

//==========================================================

void keysacn()//按键扫描

{

int val;

val=digitalRead(key);//读取数字7 口电平值赋给val

while(!digitalRead(key))//当按键没被按下时，一直循环

{

val=digitalRead(key);//此句可省略，可让循环跑空

}

while(digitalRead(key))//当按键被按下时

{

delay(10); //延时10ms

val=digitalRead(key);//读取数字7 口电平值赋给val

if(val==HIGH) //第二次判断按键是否被按下

{

digitalWrite(beep,HIGH); //蜂鸣器响

while(!digitalRead(key)) //判断按键是否被松开

digitalWrite(beep,LOW); //蜂鸣器停止

}

else

digitalWrite(beep,LOW);//蜂鸣器停止

}

}

void loop()

{

keysacn();//调用按键扫描函数

while(1)

{

//有信号为LOW 没有信号为HIGH 检测到黑线 输出高 检测到白色区域输出低

SR = digitalRead(SensorRight);//有信号表明在白色区域，车子底板上L1亮；没信号表明压在黑线上，车子底板上L1灭

SL = digitalRead(SensorLeft);//有信号表明在白色区域，车子底板上L2亮；没信号表明压在黑线上，车子底板上L2灭

if (SL == LOW&&SR==LOW)

run(); //调用前进函数

else if (SL == HIGH & SR == LOW)// 左循迹红外传感器,检测到信号，车子向右偏离轨道，向左转

spin\_left();

else if (SR == HIGH & SL == LOW) // 右循迹红外传感器,检测到信号，车子向左偏离轨道，向右转

spin\_right();

else // 都是黑色, 停止

brake();

}

}

# 超声波舵机云台避障

int Echo = A1; // Echo回声脚(P2.0)

int Trig =A0; // Trig 触发脚(P2.1)

int Front\_Distance = 0;

int Left\_Distance = 0;

int Right\_Distance = 0;

int Left\_motor=8; //左电机(IN3) 输出0 前进 输出1 后退

int Left\_motor\_pwm=9; //左电机PWM调速

int Right\_motor\_pwm=10; // 右电机PWM调速

int Right\_motor=11; // 右电机后退(IN1) 输出0 前进 输出1 后退

int key=A2;//定义按键 A2 接口

int beep=A3;//定义蜂鸣器 A3 接口

//const int SensorRight = 3; //右循迹红外传感器(P3.2 OUT1)

//const int SensorLeft = 4; //左循迹红外传感器(P3.3 OUT2)

//int SL; //左循迹红外传感器状态

//int SR; //右循迹红外传感器状态

int servopin=2;//设置舵机驱动脚到数字口2

int myangle;//定义角度变量

int pulsewidth;//定义脉宽变量

int val;

void setup()

{

Serial.begin(9600); // 初始化串口

//初始化电机驱动IO为输出方式

pinMode(Left\_motor,OUTPUT); // PIN 8 8脚无PWM功能

pinMode(Left\_motor\_pwm,OUTPUT); // PIN 9 (PWM)

pinMode(Right\_motor\_pwm,OUTPUT);// PIN 10 (PWM)

pinMode(Right\_motor,OUTPUT);// PIN 11 (PWM)

pinMode(key,INPUT);//定义按键接口为输入接口

pinMode(beep,OUTPUT);

//pinMode(SensorRight, INPUT); //定义右循迹红外传感器为输入

// pinMode(SensorLeft, INPUT); //定义左循迹红外传感器为输入

pinMode(Echo, INPUT); // 定义超声波输入脚

pinMode(Trig, OUTPUT); // 定义超声波输出脚

lcd.begin(16,2); //初始化1602液晶工作 模式

//定义1602液晶显示范围为2行16列字符

pinMode(servopin,OUTPUT);//设定舵机接口为输出接口

}

//=======================智能小车的基本动作=========================

//void run(int time) // 前进

void run() // 前进

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void brake(int time) //刹车，停车

{

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM 调速输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM 调速输出0

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100);//执行时间，可以调整

}

void brake() //刹车，停车

{

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM 调速输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM 调速输出0

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100);//执行时间，可以调整

}

void left(int time) //左转(左轮不动，右轮前进)

//void left() //左转(左轮不动，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,LOW); //左电机PWM

analogWrite(Left\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100); //执行时间，可以调整

}

void spin\_left(int time) //左转(左轮后退，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100); //执行时间，可以调整

}

void spin\_left() //左转(左轮后退，右轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机前进

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,50);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void right(int time)

//void right() //右转(右轮不动，左轮前进)

{

digitalWrite(Right\_motor,LOW); // 右电机不转

digitalWrite(Right\_motor\_pwm,LOW); // 右电机PWM输出0

analogWrite(Right\_motor\_pwm,0);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100); //执行时间，可以调整

}

void spin\_right(int time) //右转(右轮后退，左轮前进)

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机PWM输出1

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100); //执行时间，可以调整

}

void spin\_right() //右转(右轮后退，左轮前进)

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机PWM输出1

analogWrite(Right\_motor\_pwm,50);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,LOW); // 左电机前进

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

//delay(time \* 100); //执行时间，可以调整

}

void back(int time) //后退

{

digitalWrite(Right\_motor,HIGH); // 右电机后退

digitalWrite(Right\_motor\_pwm,HIGH); // 右电机前进

analogWrite(Right\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

digitalWrite(Left\_motor,HIGH); // 左电机后退

digitalWrite(Left\_motor\_pwm,HIGH); //左电机PWM

analogWrite(Left\_motor\_pwm,150);//PWM比例0~255调速，左右轮差异略增减

delay(time \* 100); //执行时间，可以调整

}

//==========================================================

void keysacn()//按键扫描

{

int val;

val=digitalRead(key);//读取数字7 口电平值赋给val

while(!digitalRead(key))//当按键没被按下时，一直循环

{

val=digitalRead(key);//此句可省略，可让循环跑空

}

while(digitalRead(key))//当按键被按下时

{

delay(10); //延时10ms

val=digitalRead(key);//读取数字7 口电平值赋给val

if(val==HIGH) //第二次判断按键是否被按下

{

digitalWrite(beep,HIGH); //蜂鸣器响

while(!digitalRead(key)) //判断按键是否被松开

digitalWrite(beep,LOW); //蜂鸣器停止

}

else

digitalWrite(beep,LOW); //蜂鸣器停止

}

}

float Distance\_test() // 量出前方距离

{

digitalWrite(Trig, LOW); // 给.触发脚低电平2μs

delayMicroseconds(2);

digitalWrite(Trig, HIGH); // 给触发脚高电平10μs，这里至少是10μs

delayMicroseconds(10);

digitalWrite(Trig, LOW); // 持续给触发脚低电

float Fdistance = pulseIn(Echo, HIGH); // 读取高电平时间(单位：微秒)

Fdistance= Fdistance/58; //为什么除以58等于厘米， Y米=（X秒\*344）/2

// X秒=（ 2\*Y米）/344 ==》X秒=0.0058\*Y米 ==》厘米=微秒/58

// Serial.print("Distance:"); //输出距离（单位：厘米）

// Serial.println(Fdistance); //显示距离

//Distance = Fdistance;

return Fdistance;

}

void Distance\_display(int Distance)//显示距离

{

if((2<Distance)&(Distance<400))

{

lcd.home(); //把光标移回左上角，即从头开始输出

lcd.print(" Distance: "); //显示

lcd.setCursor(6,2); //把光标定位在第2行，第6列

lcd.print(Distance); //显示距离

lcd.print("cm"); //显示

}

else

{

lcd.home(); //把光标移回左上角，即从头开始输出

lcd.print("!!! Out of range"); //显示

}

delay(250);

lcd.clear();

}

void servopulse(int servopin,int myangle)/\*定义一个脉冲函数，用来模拟方式产生PWM值舵机的范围是0.5MS到2.5MS 1.5MS 占空比是居中周期是20MS\*/

{

pulsewidth=(myangle\*11)+500;//将角度转化为500-2480 的脉宽值 这里的myangle就是0-180度 所以180\*11+50=2480 11是为了换成90度的时候基本就是1.5MS

digitalWrite(servopin,HIGH);//将舵机接口电平置高 90\*11+50=1490uS 就是1.5ms

delayMicroseconds(pulsewidth);//延时脉宽值的微秒数 这里调用的是微秒延时函数

digitalWrite(servopin,LOW);//将舵机接口电平置低

// delay(20-pulsewidth/1000);//延时周期内剩余时间 这里调用的是ms延时函数

delay(20-(pulsewidth\*0.001));//延时周期内剩余时间 这里调用的是ms延时函数

}

void front1\_detection()

{

//此处循环次数减少，为了增加小车遇到障碍物的反应速度

for(int i=0;i<=5;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulse(servopin,77);//模拟产生PWM

}

Front\_Distance = Distance\_test();

//Serial.print("Front\_Distance:"); //输出距离（单位：厘米）

Serial.println(Front\_Distance); //显示距离

//Distance\_display(Front\_Distance);

}

void left1\_detection()

{

for(int i=0;i<=15;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulse(servopin,175);//模拟产生PWM

}

Left\_Distance = Distance\_test();

//Serial.print("Left\_Distance:"); //输出距离（单位：厘米）

Serial.println(Left\_Distance); //显示距离

}

void right1\_detection()

{

for(int i=0;i<=15;i++) //产生PWM个数，等效延时以保证能转到响应角度

{

servopulse(servopin,5);//模拟产生PWM

}

Right\_Distance = Distance\_test();

// Serial.print("Right\_Distance:"); //输出距离（单位：厘米）

Serial.println(Right\_Distance); //显示距离Serial.read();

}

//===========================================================

void loop()

{

keysacn(); //调用按键扫描函数

while(1)

{

//SR = digitalRead(SensorRight);//有信号表明在白色区域，车子底板上L1亮；没信号表明压在黑线上，车子底板上L1灭

//SL = digitalRead(SensorLeft);//有信号表明在白色区域，车子底板上L2亮；没信号表明压在黑线上，车子底板上L2灭

front1\_detection();//测量前方距离

if(Front\_Distance <= 30)//当遇到障碍物时

{

brake(4);//先刹车

back(3);//后退减速

brake(2);//停下来做测距

left1\_detection();//测量左边距障碍物距离

Distance\_display(Left\_Distance);//液晶屏显示距离 h

right1\_detection();//测量右边距障碍物距离

Distance\_display(Right\_Distance);//液晶屏显示距离

if((Left\_Distance < 30 ) &&( Right\_Distance < 30 ))//当左右两侧均有障碍物靠得比较近

spin\_left(0.7);//旋转掉头

else if(Left\_Distance > Right\_Distance)//左边比右边空旷

{

left(7);//左转

brake(1);//刹车，稳定方向

}

else//右边比左边空旷

{

right(7);//右转

brake(1);//刹车，稳定方向

}

}

else if((Front\_Distance > 30) && (Front\_Distance < 2000))

{

run();

}

else if(Front\_Distance > 2000)

{

brake(1);//刹车，稳定方向

back(3);

left(5);

brake(2);

}

else if (SL == LOW&&SR==LOW)

run(); //调用前进函数

else if (SL == HIGH & SR == LOW)// 左循迹红外传感器,检测到信号，车子向右偏离轨道，向左转

spin\_left();

else if (SR == HIGH & SL == LOW) // 右循迹红外传感器,检测到信号，车子向左偏离轨道，向右转

spin\_right();

else // 都是黑色, 停止

brake();

}

}