

课程名称：

单片机及嵌入式系统课程设计报告

学 院 名 称 电气信息工程学院

专 业 班 级 自动化1704

学 生 学 号 3170502124

学 生 姓 名 刘佳昊

指 导 教 师 王照伟

提 交 日 期 2020.10.13

目录

**[第一章 寻迹小车设计要求 1](#_Toc31095)**

[1.1基本要求： 1](#_Toc8332)

[1.2具体要求 1](#_Toc23423)

[1.3考核目标 1](#_Toc12414)

**[第二章 课程设计题目分析及方案设计 2](#_Toc7339)**

[2.1运动控制 2](#_Toc24727)

[2.2路面黑色与非黑色信号采集 2](#_Toc23979)

[2.3有无障碍物信号采集 2](#_Toc9559)

**[第三章 硬件设计及实现 3](#_Toc1352)**

[3.1电机 3](#_Toc9608)

[3.2胡须避障 3](#_Toc5569)

[3.3红外线发射和接收器件探测道路并寻迹 5](#_Toc24221)

**[第四章 控制算法实现 8](#_Toc21028)**

[1.运动控制 8](#_Toc32044)

[2.红外检测及循迹 12](#_Toc23401)

[3.避障 15](#_Toc4644)

**[第五章 寻迹小车整体实现流程图 17](#_Toc27472)**

**[第六章 小车控制效果及结果分析 18](#_Toc21155)**

**[第七章 心得体会 19](#_Toc17240)**

**[附录：原始代码 20](#_Toc16286)**

# 寻迹小车设计要求

## 1.1基本要求：

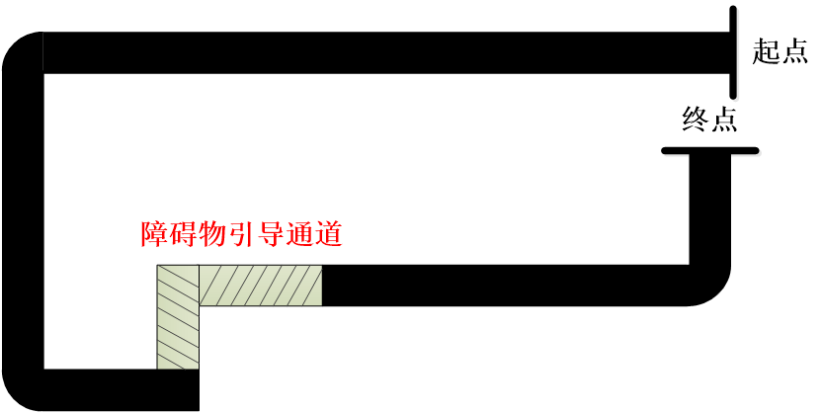
（1）机器人小车循迹（跟踪条纹带）；

（2）无条纹区动作：基于障碍物引导进行循迹导航（基于触觉的胡须避障导航，检测到障碍物，左转90°，停5秒，直行，检测到障碍物，停5秒，右转90°，直行检测到条纹，再停5秒）；

（3）全程要设计信号灯指示工作状态（启动、循迹、障碍物检测报警、停止）。

## 1.2具体要求

要求小车从起点开始，跟随黑色轨迹行进，并且能够避开中途的障碍物，跟随黑色轨迹继续行进至终点。期望小车能尽可能快的完成三圈寻迹运动，并且行进过程顺利。轨迹示例图如下：



示例图

## 1.3考核目标

考察在指定时间内学生修改软硬件的能力，具体体现为根据行走过程中的小车姿态（是否摇摆）、拐弯的顺畅性（是否偏离赛道）、对赛道的适应性（条纹带的粗细）、小车速度和拐弯慢的矛盾性以及团队协作能力等多方面的能力，提高学生对实践类课程和单片机嵌入式系统应用方面的能力。

# 第二章 课程设计题目分析及方案设计

分析题目可知，该项目旨在完成智能车循迹与避障两大功能。从控制角度出发，循迹可拆解为运动控制以及路面黑色与非黑色区分信号采集，避障可拆解为运动控制以及有无障碍物区分信号采集。由上可知，要完成智能车循迹与避障需解决运动控制，路面黑色与非黑色区分信号采集和有无障碍物区分信号采集三部分功能。以下依次讨论：

## 2.1运动控制

目前多用电机实现机器人运动控制。不同电机性能各有优劣，特点各异。针对题目给出的应用场景，智能车不但要实现左右转，调速等基本功能，还需要实现确定角度旋转等特殊要求，故而选用360度圆周舵机作为智能车驱动电机。舵机具有大扭矩，精确旋转给定角度等功能能够满足题目要求。

## 2.2路面黑色与非黑色信号采集

红外传感器常用作循迹传感器，通过发射红外线，检测返回的红外线信号强弱来判断智能车运行的路面是否为黑色。目前市面上的红外传感器从采集信号的角度大致可分为两类，一类为只能返回01数字信号的数字红外传感器，一类为可以返回0~5V模拟量的模拟红外传感器。数字红外传感器原理简单，制作成本低廉，性能上也可满足题目要求，故而采用数字红外传感器。

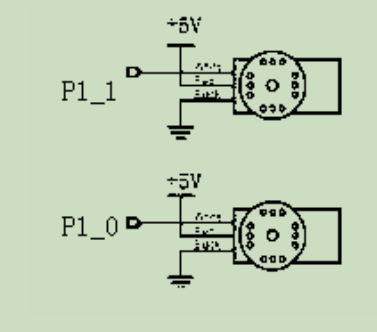
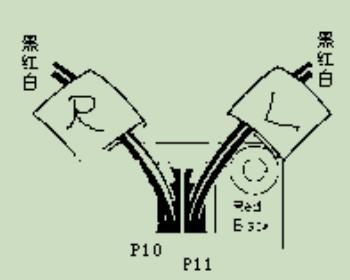
## 2.3有无障碍物信号采集

使用试验箱提供的触须和电阻等元器件可以实现简单的避障传感器。

# 第三章 硬件设计及实现

## 3.1电机

电机接线原理图如下图3-1.1所示：

3-1.1伺服电机接线原理图 3-1.2实际接线图

## 3.2胡须避障

51系列单片机有4个8位的并行I/O口：P0、P1、P2和 P3。这4个接口，既可以作为输入，也可以作为输出，既可按8位处理，也可按位方式使用。 实际上，当单片机启动或复位时，所有的I/O插脚缺省为输入。也就是说，如果将胡须 连接到单片机某个I/O管脚时，该管脚会自动作为输入。作为输入，如果I/O脚上的电压为5V， 则其相对应的I/O口寄存器中的相应位存储1；如果电压为0V，则存储0。 布置恰当的电路，可以让胡须达到上述效果：当胡须没有被碰到时，使I/O脚上的电压 为5V；当胡须被碰到时，则使I/O脚上电压为0。然后，单片机就可以读入相应数据，进行分 析、处理，控制机器人的运动。

编程让机器人通过触觉胡须导航之前，首先必须安装并测试胡须。图3.2.1所示是安装机器人触觉胡须所需的硬件元件清单，包括：

1． 金属丝2根

2． 平头M3×22盘头螺钉2个

3． 13mm圆形立柱2个

4． M3尼龙垫圈2个

5． 3-pin公-公接头2个

6． 220Ω电阻2个

7． 10kΩ电阻2个

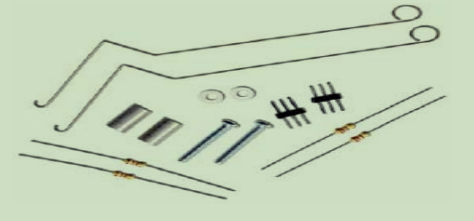
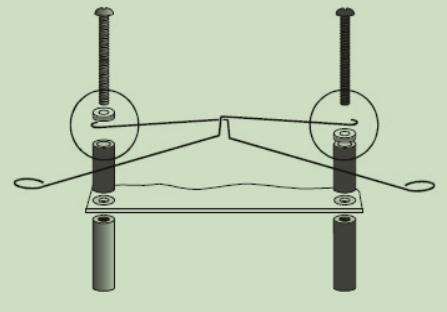


图3-2.1胡须硬件

胡须安装步骤：

①拆掉连接主板到前支架的两颗螺钉

参考图3-2.2，进行下面操作

螺钉依次穿过M3尼龙垫圈、13mm圆

形立柱

②螺钉穿过主板上的圆孔之后，拧进主板下面的支架中，但不要拧紧。

③把须状金属丝的其中一个钩在尼龙 图3-2.2胡须安装示意图

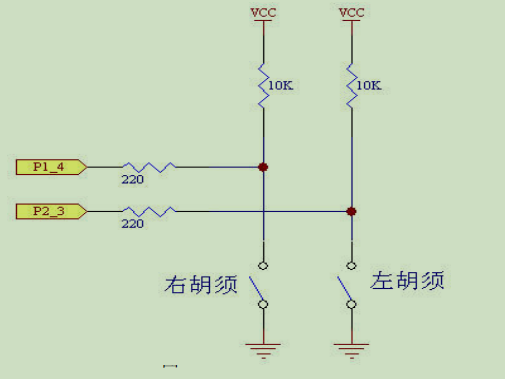
垫圈之上，另一个钩在尼龙垫圈之

下，调整它们的位置使它们横向交

叉但又不接触。

④拧紧螺钉到支架上 图4-2 安装机器人胡须

⑤ 参考接线图 图3-2.3，搭建胡须电路。

注意：右边胡须状态信息输入是通

过P1口的第4脚完成，而左边胡须

状态信息输入是通过P2口的第3脚

完成

⑥确定两条胡须比较靠近，但又不接

触面包板上的3-pin头。推荐保持3

mm的距离。

⑦ 图3-2.3所示是实际的参考接线图。

图3-2.3胡须电路示意图

## 3.3红外线发射和接收器件探测道路并寻迹

以下将介绍如何使用红外光来照射机器人前进的路线，然后确定何时有光线从被探测目标反射回来，通过检测反射回来的红外光就可以确定前方是否有物体。由于红外遥控技术的发展，现在红外线发射器和接收器已经很普及并且价格很便宜。机器人上建立的红外光探测物体系统在许多方面就象汽车的前灯系统。当汽车 前灯射出的光从障碍物体反射回来时，人的眼睛就发现了障碍物体，然后大脑处理这些信息， 并据此控制身体动作驾驶汽车。机器人使用红外线二极管LED作为前灯，如图3-3.1所示。

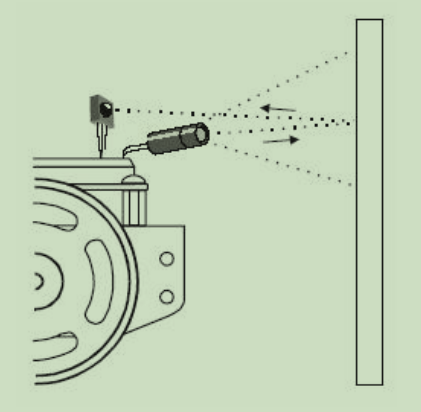
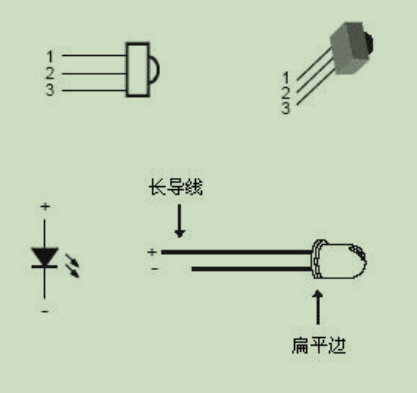
 

图3-3.1 图3-3.2

红外线（IR）接收/检测器有内置的光滤波器，除了需要检测的980 nm波长的红外线外， 它几乎不允许其它光通过。红外检测器还有一个电子滤波器，它只允许大约38.5 kHz 的电信号通过。换句话说，检测器只寻找每秒闪烁38,500次的红外光。这就防止了普通光源象太阳光和室内光对IR的干涉。太阳光是直流干涉（0Hz）源，而室内光依赖于所在区域的主电 源，闪烁频率接近100或120 Hz。由于120 Hz在电子滤波器的38.5 kHz通带频率之外，它完全被IR探测器忽略。

**元件清单:**

(1) 两个红外检测器

(2) 两个IR LED

(3) 四个470□电阻

(4) 两个9013三极管

**搭建红外线前灯**

电路板的每个角安装一个IR组(IR LED和检测器)。

① 断开主板和伺服系统的电源

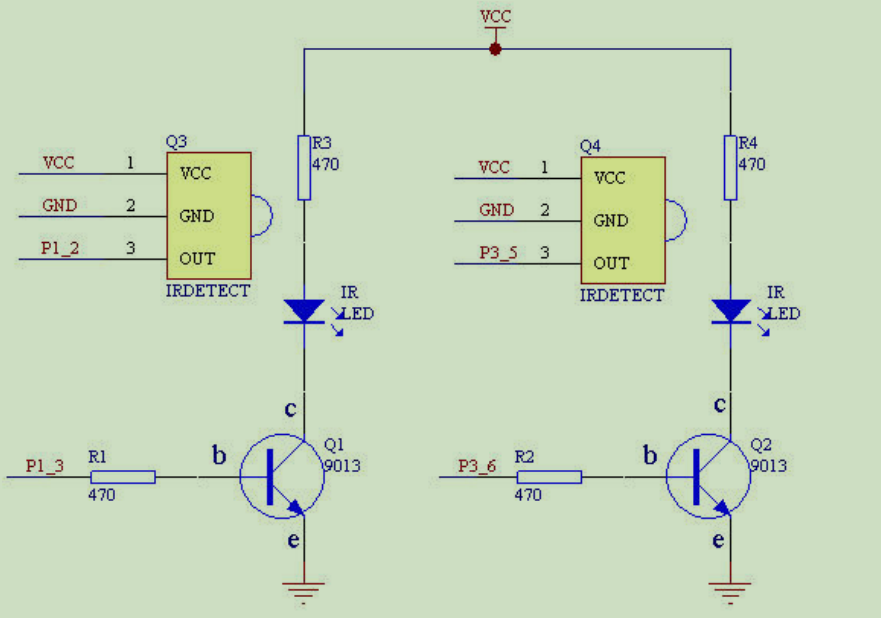
②建立图2-3.3所示的电路，

图3-3.3左侧和右侧IR组原理图

**使用三极管9013 的原因**

因为C51的IO驱动能力较弱，这里我们加入三极管使其工作在开关状态。 三极管是一种控制元件，主要用来控制电流大小，简单地说，是用小电流去控制大电流。 通过工艺的方法，把两个二极管背靠背地连接起来就组成了三极管。按 PN 结的组合方 式不同分为 PNP 型和 NPN 型。本任务中用到的是 NPN 型三极管 9013，结构示意图及符号如图3-3.4，管脚图如3-3.5。

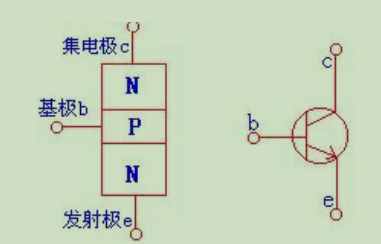
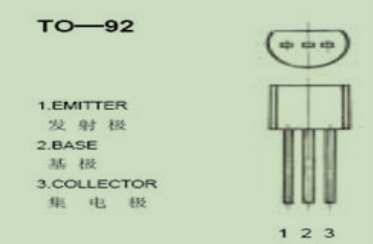
 

图3-3.4三极管结构图与符号图 图3-3.5三极管9013管脚图

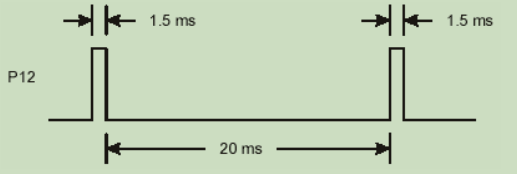
**9013 的工作原理：**它的基区做得很薄，当按图 5-3 连接时，发射结正

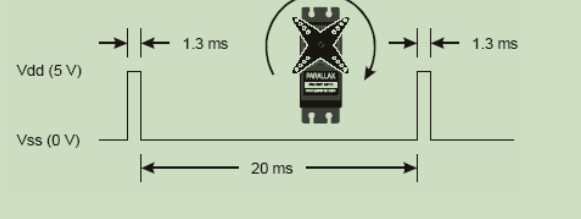
偏，集电结反偏，发射区向基区注入电子，这时由于集电结反偏，对基区的电子有很强的吸引力，所以由发射区注入基区的电子大部分进入集电区，于是集电极的电流得到了增大，三极管相当于一个开关：当 P1\_3（P3\_6）置高时，从集电区经基区到发射区电路导通，加载在 IR LED 上的电压为 VCC（5V），IR LED 向外发射红外线；当 P1\_3 （P3\_6）置低时，电路又断开，IR LED 停止发射。

# 第四章 控制算法实现

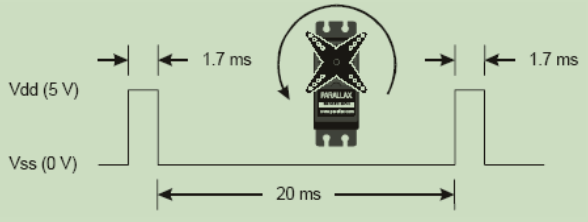
## 1.运动控制

控制伺服电机转动速度的信号如图 4-1.1、4-1.2和4-1.3所示的脉冲信号。所示是高电平持续 1.5ms 低电平持续 20ms，然后不断重复的 控制脉冲序列。该脉冲序列发给 经过零点标定后的伺服电机，伺服电机不会旋转。从图 4-1.1、4-1.2和4-1.3可知，控制电机运动转速的 是高电平持续的时间，当高电平 持续时间为 1.3ms 时，电机顺时 针全速旋转，当高电平持续时间1.7ms 时，电机逆时针速旋转。

4-1.1电动机转速为0的信号时序图



4-1.2 电动机顺时针全速转动信号时序图



4-1.3 电动机逆时针全速转动时序图

1.1单个舵机驱动实现

#define SERVOLEFTON SetLeftServo()

#define SERVOLEFTOFF ResetLeftServo()

#define SERVORIGHTON SetRightServo()

#define SERVORIGHTOFF ResetRightServo()

void SetLeftServo(void)

{

P1 |= (1<<0);

}

void ResetLeftServo(void)

{

P1 &= ~(1<<0);

}

void SetRightServo(void)

{

P1 |= (1<<1); //P1.1 = 1

}

void ResetRightServo(void)

{

P1 &= ~(1<<1); //P1.1 = 0

}

int ServoLeft(int left\_speed)

{

SERVOLEFTON;

delay\_nus(1400+left\_speed);

SERVOLEFTOFF;

delay\_nms(20); //20ms

return 0;

}

int ServoRight(int right\_speed)

{

SERVORIGHTON;

delay\_nus(1500-right\_speed);

SERVORIGHTOFF;

delay\_nms(20); //20ms

return 0;

}

1.2舵机驱动总程序

int ServoMove(int servo\_left\_speed,int servo\_right\_speed)

{

#if 1

SERVOLEFTON;

delay\_nus(1400-servo\_left\_speed);

SERVOLEFTOFF;

SERVORIGHTON;

delay\_nus(1500+servo\_right\_speed);

SERVORIGHTOFF;

delay\_nms(20);

#endif

// ServoLeft(servo\_left\_speed);

// ServoRight(servo\_right\_speed);

return 0;

}

## 2.红外检测及循迹

2.1红外检测

图4-2.1举例说明机器人如何用红外发射频率做距离测试。在这个例子中，目标物体在区域3。也就是说，发送35700Hz和38460Hz频率能发现物体，发送29370Hz、31230Hz以及33050Hz 频率就不能发现物体。如果你移动物体到区域2，那么发送33050Hz、35700Hz以及38460Hz 可以发现物体，发送29370Hz和31230Hz频率不能发现物体。

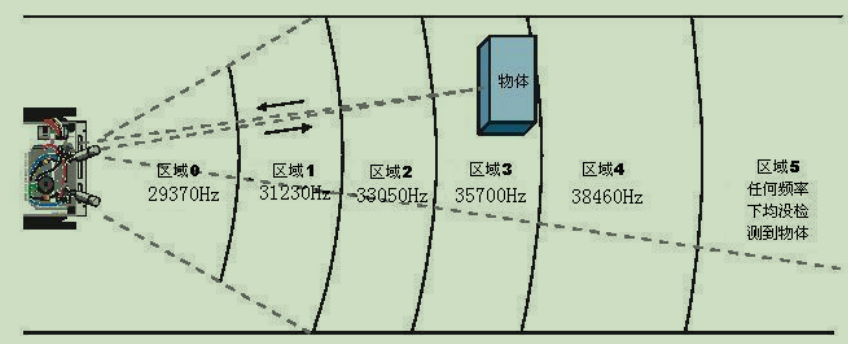


图4-2.1

现将传感器朝向地面，因黑胶带的吸光性，传感器可以测出与黑带的距离。

#define GetLeftIR ((P3&(1<<5)) != 0 ? true:false) //左边红外接收连接到P3\_5

#define GetRightIR ((P0&(1<<1)) != 0 ? true:false) //右边红外接收连接到P0\_1

//左边红外发射连接到P3\_6

#define SetLeftLaunch P3|=(1<<6) //P3.6 = 1

#define ResetLeftLaunch P3&=~(1<<6) //P3.6 = 0

//右边红外发射连接到P1\_4

#define SetRightLaunch P1|=(1<<4) //P1.4 = 1

#define ResetRightLaunch P1&=~(1<<4) //P1.4 = 0

void IRLaunch(unsigned char IR)

{

int counter;

if(IR=='L')//左边发射

for(counter=0;counter<38;counter++)

{

SetLeftLaunch;

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

ResetLeftLaunch;

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

}

if(IR=='R')//右边发射

for(counter=0;counter<38;counter++)

{

SetRightLaunch;

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

ResetRightLaunch;

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

}

}

\_bool ReadLeftSensor(void){return (\_bool)GetLeftIR;}

\_bool ReadRightSensor(void){return (\_bool)GetRightIR;}

2.2循迹

void ServoMoveFollowLine(void)

{

int irDetectLeft,irDetectRight;

SetGreenLed;

ResetRedLed;

IRLaunch('R'); //右边发射

irDetectRight = ReadRightSensor(); //右边接收

IRLaunch('L'); //左边发射

irDetectLeft = ReadLeftSensor(); //左边接收

printf("ReadLeftSensor = %d ReadRightSensor= %d\n",irDetectLeft,irDetectRight);

if((irDetectLeft==1)&&(irDetectRight==1)) { ServoMove(250,250); }

else if((irDetectLeft==1)&&(irDetectRight==0)) { ServoMove(250,-200); } //turn right

else if((irDetectLeft==0)&&(irDetectRight==1)) { ServoMove(-200,250); } //turn left

else { ServoMove(-250,250);}

}

3.避障

通过采集指定I/O口数字量作为触发信号，接受到触发信号之后，智能车启动避障功能。

if((ReadLeftBeard() == 0)||ReadLeftBeard()==0)

{

if(!BEARDTIME)

{

ResetRedLed;

ResetGreenLed;

for(i = 0;i<26;i++){ ServoMove(-250,250); }

BEARDTIME++;

ServoMove(1,1);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

}

}

if((ReadLeftBeard() == 0)||ReadLeftBeard()==0)

{

if(BEARDTIME)

{

ResetRedLed;

ResetGreenLed;

for(j = 0;j<26;j++){ ServoMove(250,-250);

}

BEARDTIME=0;}

ServoMove(1,1);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

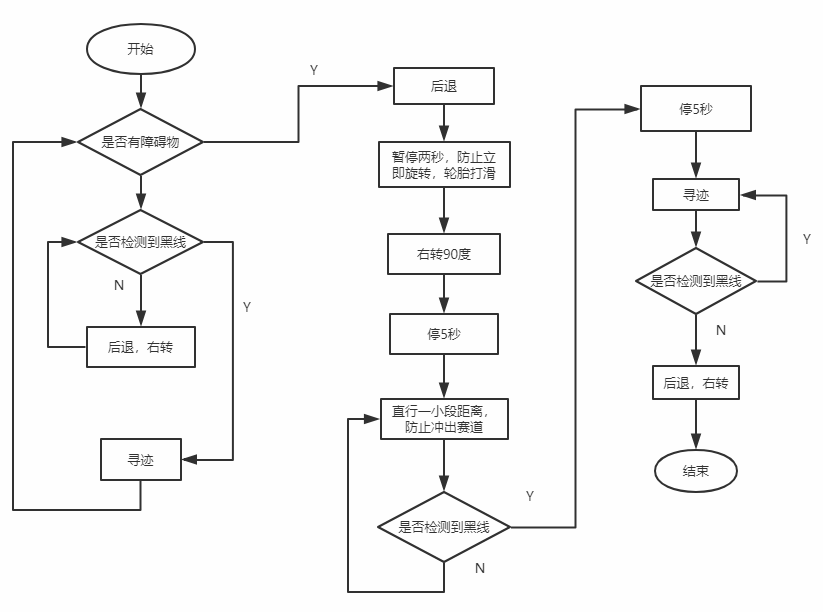
delay\_nms(1000);

BEARDTIME=0;

}

# 寻迹小车整体实现流程图

图5-1是整体程序流程框图。

图5-1

# 小车控制效果及结果分析

小车可以实现以下功能：

（1）机器人小车循迹（跟踪条纹带）；

（2）无条纹区动作：基于障碍物引导进行循迹导航（基于触觉的胡须避障导航，检测到障碍物，左转90°，停5秒，直行，检测到障碍物，停5秒，右转90°，直行检测到条纹，再停5秒）；

（3）全程设计信号灯指示工作状态（启动、循迹、障碍物检测报警、停止）。

整个程序采用松耦合，模块化的设计，在程序节点处多设置了日志打印，能够很快的完成整个系统的开发。

# 心得体会

时间匆匆流逝，作为大四的“老人”，看着手里的循迹小车，感慨万千。遥想大一刚来时，就是因为416的循迹小车比赛，将我带入嵌入式的世界。后来的很多日子，数不清有多少个夜晚为了准备比赛熬夜写代码，查资料。现在，嵌入式软件开发也成为我将来谋生的手段。暑假期间，当第一次以一名嵌入式软件工程师的身份参加工作的时候，由于本科阶段的实践经验，所以我可以很快的跟上公司的开发节奏。这几天的课程设计，用的是我较早接触到的一款mcu，整体难度开发不大，但对我而言还是有所突破的，比如C语言的代码规范和松耦合，模块化的架构设计。嵌入式的路还很长，裸机，RTos，linux每一个方向都还有很多的东西要学。感谢学校提供的平台，让我得以遇见乐意奋斗终生的方向。

# 附录：原始代码

/\*

\*main.c

\*/

#include "main.h"

#include "delay.h"

#include "servo.h"

#include "uart.h"

#include "sensor.h"

extern int BEARDTIME;

int main(void)

{

int i = 0,j = 0;

//启动

SetGreenLed;

SetRedLed;

#if (UARTON == 1)

uart\_init();

#endif

BEARDTIME = 0;

while(1)

{

// ServoMove(200,200);

ServoMoveFollowLine();

if((ReadLeftBeard() == 0)||ReadLeftBeard()==0)

{

if(!BEARDTIME)

{

ResetRedLed;

ResetGreenLed;

for(i = 0;i<26;i++){ ServoMove(-250,250); }

BEARDTIME++;

ServoMove(1,1);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

}

}

if((ReadLeftBeard() == 0)||ReadLeftBeard()==0)

{

if(BEARDTIME)

{

ResetRedLed;

ResetGreenLed;

for(j = 0;j<26;j++){ ServoMove(250,-250); }

BEARDTIME=0;}

ServoMove(1,1);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

BEARDTIME=0;

}

}

return 0;

}

/\*

\*sensor.c

\*/

#include "main.h"

#include "intrins.h"

#include "sensor.h"

void IRLaunch(unsigned char IR)

{

int counter;

if(IR=='L')//左边发射

for(counter=0;counter<38;counter++)

{

SetLeftLaunch;

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

ResetLeftLaunch;

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

}

if(IR=='R')//右边发射

for(counter=0;counter<38;counter++)

{

SetRightLaunch;

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

ResetRightLaunch;

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

\_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_(); \_nop\_();

}

}

\_bool ReadLeftSensor(void){return (\_bool)GetLeftIR;}

\_bool ReadRightSensor(void){return (\_bool)GetRightIR;}

\_bool ReadLeftBeard(void){return (\_bool)GetLeftBeard;}

\_bool ReadRightBeard(void){return (\_bool)GetRightBeard;}

/\*

\*senvo.c

\*/

#include "main.h"

#include "delay.h"

#include "servo.h"

#include "sensor.h"

#define SERVOLEFTON SetLeftServo()

#define SERVOLEFTOFF ResetLeftServo()

#define SERVORIGHTON SetRightServo()

#define SERVORIGHTOFF ResetRightServo()

int BEARDTIME = 0;

int ServoLeft(int left\_speed)

{

SERVOLEFTON;

delay\_nus(1400+left\_speed);

SERVOLEFTOFF;

delay\_nms(20); //20ms

return 0;

}

int ServoRight(int right\_speed)

{

SERVORIGHTON;

delay\_nus(1500-right\_speed);

SERVORIGHTOFF;

delay\_nms(20); //20ms

return 0;

}

int ServoMove(int servo\_left\_speed,int servo\_right\_speed)

{

#if 1

SERVOLEFTON;

delay\_nus(1400-servo\_left\_speed);

SERVOLEFTOFF;

SERVORIGHTON;

delay\_nus(1500+servo\_right\_speed);

SERVORIGHTOFF;

delay\_nms(20);

#endif

// ServoLeft(servo\_left\_speed);

// ServoRight(servo\_right\_speed);

return 0;

}

void SetLeftServo(void)

{

P1 |= (1<<0);

}

void ResetLeftServo(void)

{

P1 &= ~(1<<0);

}

void SetRightServo(void)

{

P1 |= (1<<1); //P1.1 = 1

}

void ResetRightServo(void)

{

P1 &= ~(1<<1); //P1.1 = 0

}

void ServoMoveFollowLine(void)

{

int irDetectLeft,irDetectRight;

SetGreenLed;

ResetRedLed;

IRLaunch('R'); //右边发射

irDetectRight = ReadRightSensor(); //右边接收

IRLaunch('L'); //左边发射

irDetectLeft = ReadLeftSensor(); //左边接收

printf("ReadLeftSensor = %d ReadRightSensor= %d\n",irDetectLeft,irDetectRight);

// if((irDetectLeft==0)&&(irDetectRight==0)) { ServoMove(200,200); } //IR传感器

// else if((irDetectLeft==0)&&(irDetectRight==1)) { ServoMove(200,-100); } //turn right

// else if((irDetectLeft==1)&&(irDetectRight==0)) { ServoMove(-100,200); } //turn left

// else { ServoMove(-100,200);} //

if((irDetectLeft==1)&&(irDetectRight==1)) { ServoMove(250,250); } //光电管

else if((irDetectLeft==1)&&(irDetectRight==0)) { ServoMove(250,-200); } //turn right

else if((irDetectLeft==0)&&(irDetectRight==1)) { ServoMove(-200,250); } //turn left

else { ServoMove(-250,250);} //

}

void test\_sesor(void)

{

int irDetectLeft,irDetectRight;

IRLaunch('R'); //右边发射

irDetectRight = ReadRightSensor(); //右边接收

IRLaunch('L'); //左边发射

irDetectLeft = ReadLeftSensor(); //左边接收

printf("ReadLeftSensor = %d ReadRightSensor= %d\n",irDetectLeft,irDetectRight); //

}

void ServoMoveFollowBeard(void)

{

int BeardDetectLeft,BeardDetectRight;

BeardDetectLeft = ReadLeftBeard();

BeardDetectRight = ReadRightBeard();

if((BeardDetectLeft == 0)||BeardDetectRight==0){ BEARDTIME++; }

if(1 == BEARDTIME){ServoMove(0,200);delay\_nms(500);}

if(2 == BEARDTIME){ServoMove(200,0);delay\_nms(500);}

}

void bread\_touch\_1(int a,int b)

{

int i=0;

if((a == 0)||b == 0)

{

if(!BEARDTIME)

{

ResetGreenLed;

SetRedLed;

for(i = 0;i<26;i++){ ServoMove(-250,250); }

BEARDTIME++;

ResetRedLed;

ResetGreenLed;

ServoMove(1,1);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

}

}

}

void bread\_touch\_2(int a,int b)

{

int j=0;

if((a == 0)||b ==0 )

{

if(BEARDTIME)

{

ResetGreenLed;

SetRedLed;

for(j = 0;j<26;j++){ ServoMove(250,-250); }

BEARDTIME=0;

ServoMove(1,1);

ResetRedLed;

ResetGreenLed;

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

delay\_nms(1000);

BEARDTIME++;

}

}

}

void bread\_touch\_3(int a,int b)

{

if((a == 0)||b ==0 )

{

if(BEARDTIME==2)

{

ResetGreenLed;

ResetRedLed;

ServoMove(1,1);

BEARDTIME=0;

while(1){;}

}

}

}