**目 录**

**[一、 理论分析与计算 3](#_Toc17168)**

**[1. 有限状态机原理 3](#_Toc15765)**

**[2. 解迷宫法则 4](#_Toc25673)**

**[2.1 左转法则 4](#_Toc3668)**

**[2.2 右转法则 4](#_Toc21233)**

**[2.3 特殊法则 5](#_Toc9549)**

**[二、 电路与程序设计 5](#_Toc21587)**

**[1. 控制电路原理框图 5](#_Toc225)**

**[2. 控制程序流程图 7](#_Toc2692)**

**[3. 程序清单 8](#_Toc19239)**

**[三、 测试方案与测试结果 9](#_Toc3658)**

**[1. 测试方案 9](#_Toc7340)**

**[1.1 对传感器在赛道上出现的情况进行测试 9](#_Toc30043)**

**[1.2 对设计的走迷宫法则进行测试 9](#_Toc32318)**

**[1.3 对小车不同运动状态的轮速和延时进行测试 9](#_Toc3952)**

**[2. 测试结果 9](#_Toc27192)**

**[2.1 传感器在赛道上情况的测试结果 9](#_Toc14585)**

**[2.2 走迷宫法则测试结果 10](#_Toc20698)**

**[2.3 小车不同运动状态轮速和延时的测试结果 10](#_Toc22652)**

**[3. 测试结果分析 11](#_Toc181)**

**一、理论分析与计算**

1. **有限状态机原理**

有限状态机，是指某种事物有限的状态与其触发的事件和状态之间转换的数学模型。在我们的小车巡线中，有限状态机就是小车红外传感模块与其运动状态之间的关系。在本次的迷宫中，道路对应状态如下所示：

|  |  |  |
| --- | --- | --- |
| 道路情况 | 传感器对应的二进制状态 | 小车运动状况 |
| 直行道路 | 0001 1000 | 直行 |
| 左转道路 | 1111 1000 | 左转 |
| T字入口 | 1111 1111 | 左转或右转 |
| 十字入口 | 1111 1111 | 左转或右转或直行 |
| 右转道路 | 0001 1111 | 右转 |
| 直行左转道路 | 1111 1000 | 直行或左转 |
| 直行右转道路 | 0001 1111 | 直行或右转 |
| 终点 | 1101 1011 | 停止 |
| 无路径 | 0000 0000 | 原地旋转 |

上表列出了最简单的几种情况，实际跑迷宫的状态更多，需要添加更多的元素和组合。而且，上表有些状态可以有多个运动状态与之对应，实际上是不可取的，这样会跑不出迷宫，需要根据实际情况选取运动状态。

1. **解迷宫法则**

为了让小车能够根据自己的识别走出迷宫，就需要制定相应的解迷宫法则，否则可能会在一种路况下有多种选择，导致小车困在迷宫之中。下面是我们制定的几种法则：

* 1. **左转法则**

遇到能够左转的路口一律左转，遇到直行右转的路口一律直行（该决策是根据迷宫具体路况决定的，即只有这样配合左转法则才能走完迷宫），具体对应情况如下表所示：

|  |  |
| --- | --- |
| 道路情况 | 小车运动状况 |
| 直行道路 | 直行 |
| 左转道路 | 左转 |
| T字入口 | 左转 |
| 十字入口 | 左转 |
| 右转道路 | 右转 |
| 直行左转道路 | 左转 |
| 直行右转道路 | 直行 |
| 终点 | 停止 |
| 无路径 | 原地旋转 |

* 1. **右转法则**

遇到能够右转的路口一律右转，遇到直行左转的路口一律直行（该决策是根据迷宫具体路况决定的，即只有这样配合右转法则才能走完迷宫），具体对应情况如下表所示：

|  |  |
| --- | --- |
| 道路情况 | 小车运动状况 |
| 直行道路 | 直行 |
| 左转道路 | 左转 |
| T字入口 | 右转 |
| 十字入口 | 右转 |
| 右转道路 | 右转 |
| 直行左转道路 | 直行 |
| 直行右转道路 | 右转 |
| 终点 | 停止 |
| 无路径 | 原地旋转 |

**2.3 特殊法则**

根据迷宫的道路情况，制定出最短的路线，相应的路况选择不一样运动状态，具体对应情况如下表所示：

|  |  |
| --- | --- |
| 道路情况 | 小车运动状况 |
| 直行道路 | 直行 |
| 左转道路 | 左转 |
| 右转道路 | 右转 |
| 终点 | 停止 |
| 无路径 | 原地旋转 |
| 第一个十字入口 | 直行 |
| 第一个T字入口 | 左转 |
| 第一个直行左转道路 | 左转 |
| 第二个十字入口 | 右转 |
| 第三个十字入口 | 右转 |
| 第一个直行右转道路 | 直行 |
| 第四个十字入口 | 左转 |
| 第五个十字路口 | 右转 |
| 第二个直行左转入口 | 直行 |

以上三种法则都是理想状态下设计的法则，具体实现需要根据测试状况来选择跑的最好的一种方法。

**二、电路与程序设计**

1. **控制电路原理框图**

红外传感模块：

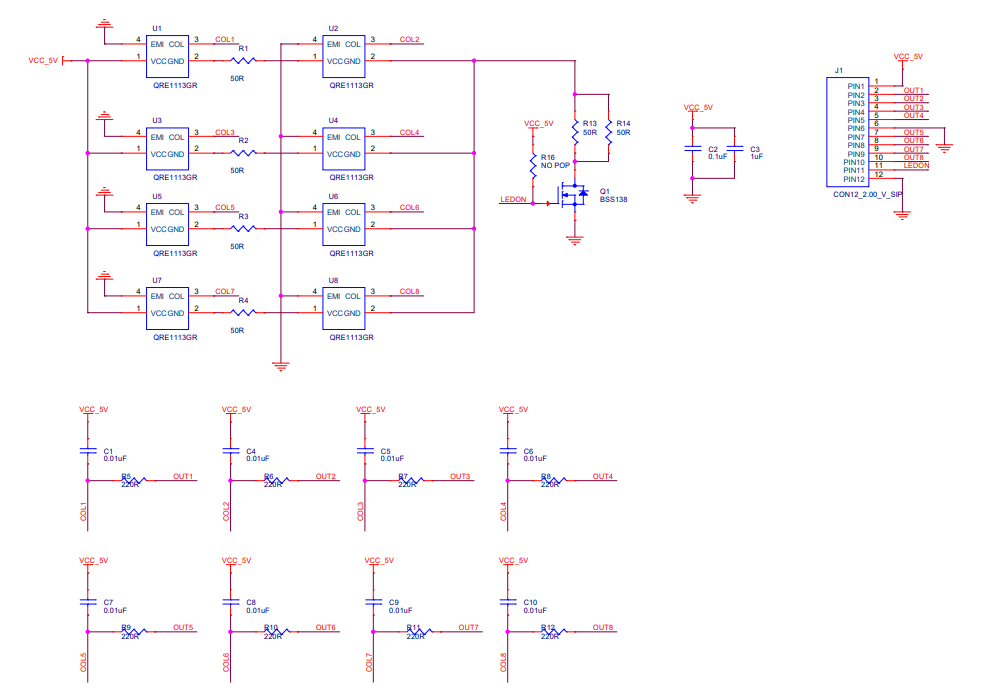


图1 红外传感模块电路原理图

电机驱动模块：

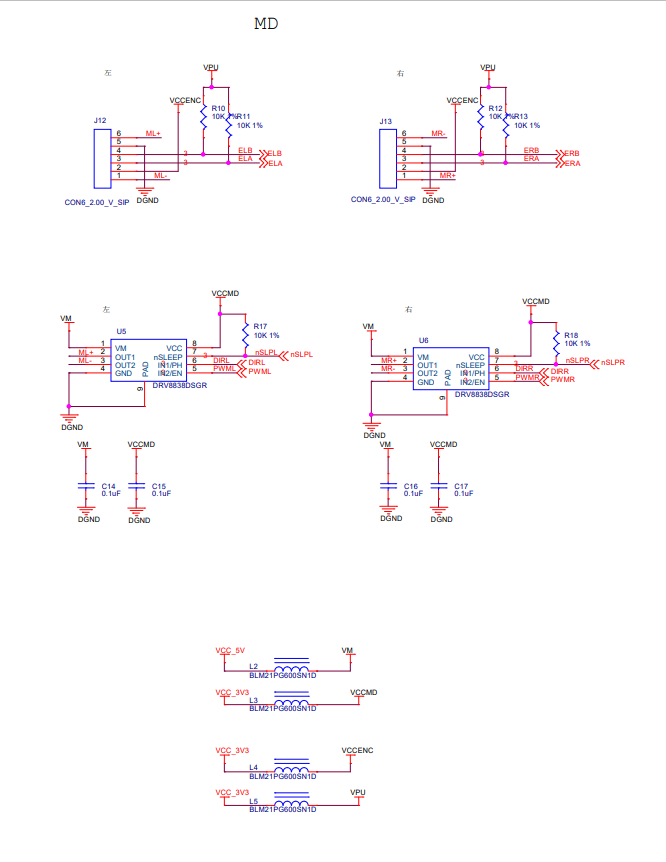
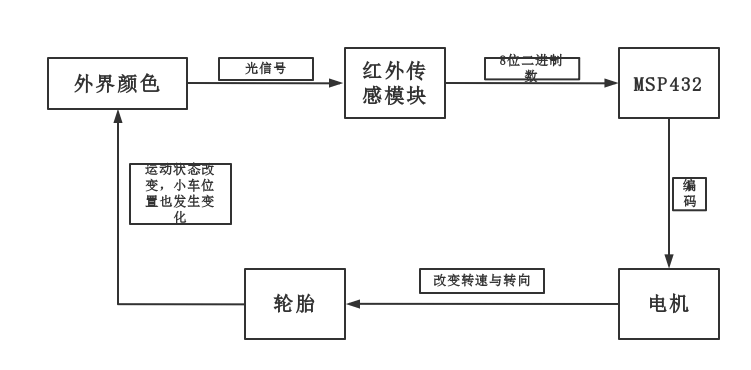


图2 电机驱动电路原理图

控制原理框图：



图三 控制原理框图

1. **控制程序流程图**

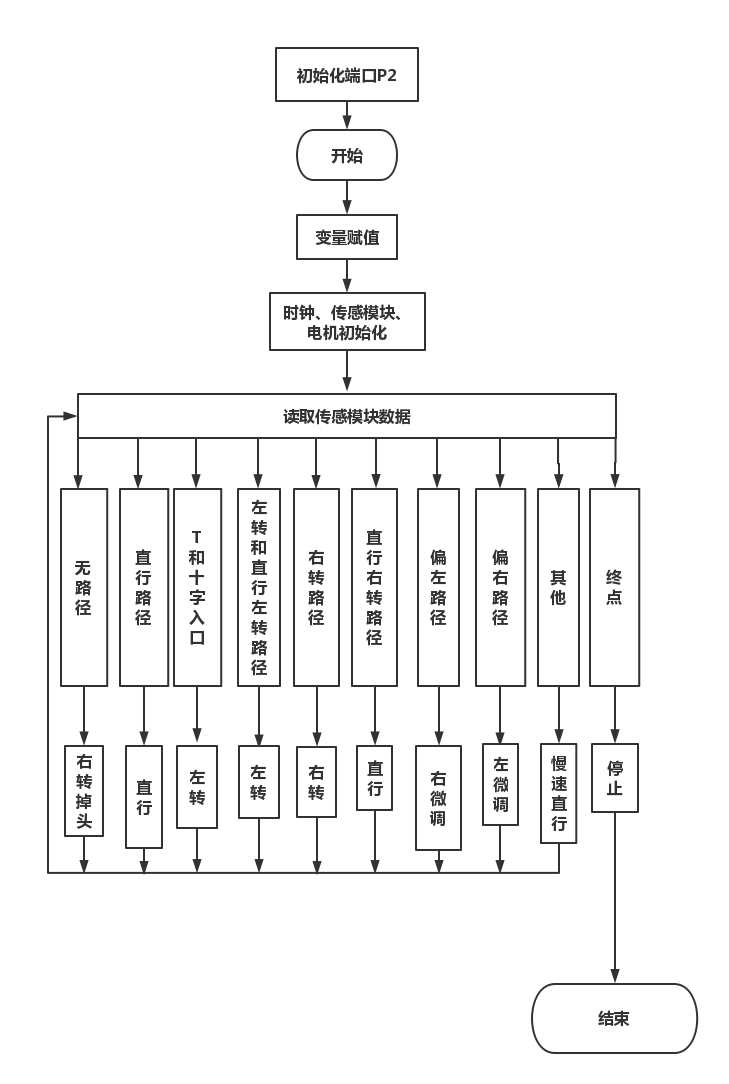


图4 程序流程图

1. **程序清单**

见附件

**三、测试方案与测试结果**

1. **测试方案**
   1. **对传感器在赛道上出现的情况进行测试**

首先，先写出传感器的基本九种情况（如一、1的表所示）；然后用USB线连接小车与电脑，在电脑终端显示出传感器的状态，并让小车在赛道上运动；若小车在赛道上出现偏离或者在预计需要改变运动状态的地方没有改变运动状态，记录下此刻终端上传感器的情况，将此情况加入到对应预期的运动状态中；如此重复使小车可以跑完赛道，确定所有的传感器状态与小车运动状态的对应情况。

* 1. **对设计的走迷宫法则进行测试**

在测试完第一步后，将上面（一、2）的三种走迷宫法则应用到小车程序中，控制其他参数一样，观察哪几种法则可以跑完迷宫，哪种能够跑的更快。

* 1. **对小车不同运动状态的轮速和延时进行测试**

在确定完跑迷宫最优的法则后，改变小车轮速的参数与相应运动状态的延时参数，将小车放入赛道中，观察其是否能稳定地跑完全程；在可以跑完迷宫的基础上，确定速度快且稳定的最优参数。

1. **测试结果**

由于我们小车的U1传感器发生了故障，故我们在测试的时候实际的结果都是坏了一个传感器的情况。

* 1. **传感器在赛道上情况的测试结果**

|  |  |  |
| --- | --- | --- |
| 道路情况 | 传感器对应的二进制状态 | 小车运动状况 |
| 直行道路 | 0001 1000 | 直行 |
| 左转和直行左转道路 | 1111 0000  1111 1000  0111 1000 | 左转 |
| T和十字入口 | 1111 1111  1111 1110  0111 1111  0111 1110 | 左转 |
| 右转道路 | 0001 1111  0000 1111  0001 1110 | 右转 |
| 直行右转道路 | 0001 1000  0000 1000  0000 0100  0000 1100  0001 0000  0010 0000  0011 0000 | 直行 |
| 终点 | 1101 1010  1001 1000  1011 1000  1011 1010  1011 0110  1110 1100  0110 1100 | 停止 |
| 无路径 | 0000 0000 | 原地旋转 |
| 偏左 | 0000 1000  0000 1100 | 右微调 |
| 偏右 | 0001 0000  0011 0000 | 左微调 |
| 其他 | 其他 | 慢速直行 |

* 1. **走迷宫法则测试结果**

经过测试，左手法则（一、2.1）和右手法则（一、2.2）都可以完成迷宫而特殊法则（一、2.3）不能完成迷宫；在保证稳定的情况下，左手法则完成平时时间比右手法则少。

* 1. **小车不同运动状态轮速和延时的测试结果**

经过测试，小车直行的轮速设置为（5000，5000）；左右偏的轮速设置为（5000，1000）、（1000，5000）；左右转的轮速设置为（5000，6000）、（6000，5000），延时设置为325ms；右转掉头延时设置为350ms最为稳定且速度快。

1. **测试结果分析**

对于传感器的情况，我们的小车U1传感器有故障；但在测试过程中，由于其是最边上的传感器，在赛道上几乎不会有黑线跟其有关，因此实际上其对小车的运动状态和路况决策几乎不会有影响。

对于走迷宫法则，特殊法则理论上是可行的而且能达到走完全程的速度最快，但在测试中其并不能顺利完成，原因是容错率太低了。左右手法则也不能每一次都可以顺利跑完，但容错率高，跑错了可以自行拯救；而特殊法则是设置好在第几个路口要变为什么运动状态，其中有一个误识别就会导致整个流程的失败且无法挽回。

对于轮速和延时的测试，不同的小车可以试出不同的结果，因为每辆小车的电机控制有误差，轮胎松紧也不一样。对于特定的小车，该测试就是一个不断试错和调参的过程。我们得到的数据也不一定是最优解，但却是在有限时间下我们能得到的最好的解。