

第十七届全国大学生

智能汽车竞赛

# 技 术 报 告



学 校： 河南理工大学

队伍名称： HPU\_OS

参赛队员： 常炳星

赵国浩

席 文

乔笑翊

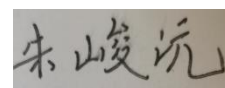
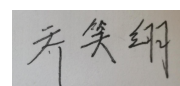
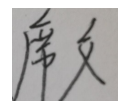
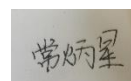
朱峻沅

带队教师： 张 丽      王 静

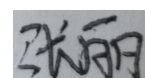
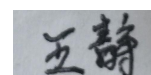
## 关于技术报告和学术论文使用授权的说明

本人完全了解全国大学生智能汽车竞赛关保留、使用技术报告和学术论文的规定，即：参赛作品著作权归参赛者本人，比赛组委会可以在相关主页上收录并公开参赛作品的设计方案、技术报告以及参赛模型车的视频、图像资料，并将相关内容编纂收录在组委会出版论文集中。

参赛队员签名：



指导老师签名：



## 目录

摘 要 .....	4
第一章 引 言 .....	5
第二章 完全模型组规则介绍 .....	7
第三章 智能车机械的结构调整与优化 .....	11
3.1 智能车摄像头位置以及安装固定 .....	11
3.2 车模内空间安排 .....	12
3.3 自制车壳的设计和打印 .....	12
3.4 前桥以及重心位置调整 .....	12
第四章 硬件电路设计 .....	13
4.1 硬件方案设计 .....	13
4.2 供电管理 .....	13
4.3 主控电路和驱动电路 .....	14
4.4 总结 .....	15
第五章 智能车控制设计说明 .....	16
5.1 赛道识别算法实现 .....	17
5.1.1 图像预处理 .....	17
5.1.2 基本扫线 .....	17
5.2.1 车库识别 .....	18
5.2.2 三叉识别 .....	18
5.2.3 十字识别 .....	19
5.2.4 圆环识别 .....	19
5.2.5 直道判断 .....	20
5.4 控制算法实现 .....	21
5.4.1 模拟式 PID 调节器 .....	21
5.4.2 数字式 PID 调节器 .....	22
5.5 电机的控制 .....	24
5.6 舵机的控制 .....	25
5.7 速度的控制 .....	25
第六章 系统开发与调试 .....	26
6.1 下位机程序开发工具 .....	26
6.2 上位机开发工具 .....	26
6.2.1 上位机编写代码 .....	26
6.2.2 下位机屏幕调参 .....	27
第七章 车模参数 .....	28
2.信息标记在 pcb 开窗层、绿油层 (solder mask) .....	30
第八章 参赛心得体会 .....	31
参考文献 .....	32
附 录 .....	33

## 摘 要

本文介绍了河南理工大学完全模型组“HPU\_OS”团队在第十七届全国大学生智能汽车竞赛中的设计方案。本方案采用组委会规定的专为完全模型组设计的 I 型车模，以 EdgeBoard 板卡作为上位机进行车模的赛道元素检测识别，以英飞凌 TC264 芯片作为下位机进行车模运动控制，模型算法使用百度 Paddle 框架搭建，采用部署在 EdgeBoard 板卡上的 USB 彩色摄像头来获取赛道上黑色边界线以及固定地标元素的信息；另外通过编码器检测电机实时速度，使用 PID 算法对电机和舵机进行闭环控制。为了提高模型车的速度和稳定性，使用上位机、按键、显示模块等调试工具，进行了大量硬件与软件测试。本文从比赛规则开始介绍，介绍了本系统的硬件设计、软件设计、调试方法，为大家呈现出本系统的各项设计方案。

**关键词：**智能汽车竞赛 EdgeBoard 英飞凌 TC264 Paddle 框架 Opencv PID 算法

# 第一章 引言

## 1.1 智能汽车大赛介绍

本文以第十七届全国大学生智能汽车竞赛为背景，该比赛受教育部高等教育司委托，由教育部高等自动化专业教学指导分委员会（以下简称自动化分教指委）主办全国大学生智能汽车竞赛。该竞赛以智能汽车为研究对象的创意性科技竞赛，是面向全国大学生的一种具有探索性工程实践活动，是教育部倡导的大学生科技竞赛之一，为加强大学生实践、创新能力和团队精神的培养，促进高等教育教学改革。该竞赛以“立足培养，重在参与，鼓励探索，追求卓越”为指导思想，旨在促进高等学校素质教育，培养大学生的综合知识运用能力、基本工程实践能力和创新意识，激发大学生从事科学研究与探索的兴趣和潜能，倡导理论联系实际、求真务实的学风和团队协作的人文精神，为优秀人才的脱颖而出创造条件。该竞赛由竞赛秘书处为各参赛队提供/购置规定范围内的标准硬软件技术平台，竞赛过程包括理论设计、实际制作、整车调试、现场比赛等环节，要求学生组成团队，协同工作，初步体会一个工程性的研究开发项目从设计到实现的全过程。该竞赛融科学性、趣味性和观赏性为一体，是以迅猛发展、前景广阔的汽车电子为背景，涵盖自动控制、模式识别、传感技术、电子、电气、计算机、机械与汽车等多学科专业的创意性比赛。该竞赛规则透明，评价标准客观，坚持公开、公平、公正的原则，保证竞赛向健康、普及，持续的方向发展。

据我了解，智能汽车比赛中，智能汽车就是在原有车辆系统的基础上增加了计算机处理系统、摄像头和一些传感器实现智能化，另外近几年随着深度学习的发展，如何让机器通过自己学习实现部分功能成为一大议题，卷积神经网络可以从大量数据中学习数据的特征部分，进而可以在未知复杂的场景下进行运用与具体功能实现。近期火爆的无人驾驶技术便依赖以上提到的人工智能与视觉部分技术。

## 1.2 智能汽车制作情况概述

根据竞赛规则及功能要求，本系统将在完全模型官方提供的 I 车模以及配套电机舵机以及传动系统的车模平台基础上，使用配套 EdgeBoard 计算卡和英飞凌芯片 TC264 作为核心微控制器，以及使用指定 Paddle 框架搭建模型算法，在 AI Studio 平台学习模型和训练模型，采用 USB 免驱彩色摄像头直连 EdgeBoard 计算卡用于赛道及其元素的检测，软件开发工具为 C/C++ 语言、python 语言进行软件开发构建完整的小车系统。小车通过调用 Opencv 库读取摄像头采集到的赛道信息送入 EdgeBoard 计算卡中，在计算卡中对输入的原始图像信息进行多线程操作，分别将

图像用于模型识别与巡线处理中，进而分析结果进行赛道类型决策，进行据此选择最优行进路线并进行速度的控制。整个系统设计包括车体机械结构设计和软/硬件系统设计、车体机械结构设计主要包括前后轮的调节、PCB 板的布局、车身底盘的改装、图像传感器、舵机以及编码器的安装等；

下面分别通过整体方案概述、问题描述、技术方案、方案实现、测试分析、作品总结等几个方面进行具体阐述。

本报告将从硬件到软件一一的为大家呈现该系统的设计方案和制作过程。

## 第二章 完全模型组规则介绍

### 2.1 比赛场地

#### 2.1.1 比赛环境

赛道环境与竞速室内赛道相同

#### 2.1.2 比赛场地

##### (1) 比赛场地

完全模型组比赛赛道以室内循环赛道为基础，赛道材质，赛道规格均保持一致。在导引方式上完全保留室内循环赛道的导引方式，并在此基础上添加完全模型组任务导引标志和锥桶，引导车模完成完全模型组赛道任务。

##### (2) 赛道标志

为了引导比赛任务的完成，在比赛赛道的任务元素和特殊元素区域的前方指

序号	名称	说明	图示
1	泛行区标志	表示前方三岔路口围成的泛行区域，内部区域包括蓝色底布均可行驶。	
2	禁止通行标志	放置在泛行区域进出口连线上，车辆需要绕过此标志进行通行。（此标志高出距离地面有 2cm 高度其余均紧贴）。	
3	施工区标志	表示前方为施工区，需要绕行赛道外障碍桩围成的临时路段。	
4	坡道标志	表示道路前方有坡道。	
5	加油站标志	表示前方为加油站，车辆需要驶入加油站并按照指定的出口驶出加油站。	
6	加油站出口数字标志	加油站设置有“1”和“2”两个出口，并在出口地面贴有对应的“1”和“2”数字标志。比赛时加油站的入口处会随机放置车辆需要驶出时的出口数字。	

定的区域贴有固定的地面标志。标志的样式和含义如下表所示：

▲ 图 2.1.1 赛道标志

标志的整体外框尺寸为 16cm×16cm 的正方形，标志颜色为红色，正方形内

多余的灰色区域裁减掉，国赛采用可移除性不干胶制作，按照赛道任务和元素贴在赛道的指定位置（详见任务说明）。组委会提供标准的标志 PDF 版附件（附有尺寸），参赛队可以自行打印使用。

### （3）任务锥桶

比赛在基础赛道外设置有脱离基础赛道的任务区域，这些任务区域由可以移



动的锥桶在基础赛道的边缘临时搭建而成。

▲ 图 2.1.2 锥桶

锥桶由塑料材质制作而成，外表面为红色纯色，无任何标志。锥桶的底部直径 74mm，高度 74mm。在赛道搭建的过程中锥桶为可移动状态，车辆在运行过程中触碰到锥桶，导致锥桶偏离所在位置时，车模按冲出赛道处理。

## 2.2 比赛任务

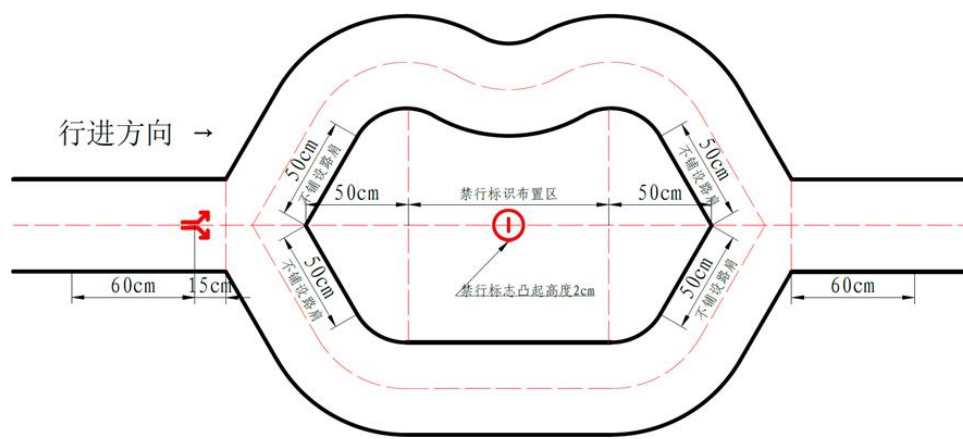
完全模型组的基本比赛任务为：

选手制作的车模完成从车库出发沿着赛道运行两周。车模需要分别通过道路设置的各种元素，识别道路中心的标志完成特殊路段通行。

比赛时间从车模驶出车库到重新回到车库为止。如果车模没有能够停止在车库内停车区内，比赛时间加罚 5 秒钟。对于未完成任务会通过相应的加罚时间叠加在比赛时间上。

### 2.2.1 泛行区

两个岔路口围成的区域为泛行区，在完全模型组别中小车行驶不受三岔路两条线路内部闭合的道路边界线的限制，内部闭合的道路边界线特定区间内不铺设路肩。参赛队自行选择最优的路线从三岔路的入口行驶到出口即可。三岔路口的





### 2.2.2 施工区

▲ 图 2.2.2 施工区

### 2.2.3 加油站

[illegible]

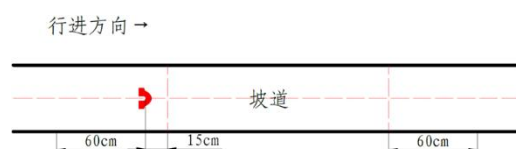
两圈的数字不相同。车辆识别到数字后驶出加油站，当车模离开加油站之后，再由裁判员将标牌更换。加油站区域前方 70 厘米处赛道中心贴有加油站标志，指示车辆前方为加油站需要进入加油站后驶出。加油站任务元素仅在全国赛上使用。

▲ 图 2.2.3 加油站

图中每个红色圆圈代表 1 个小锥桶，实际摆设中存在锥桶中心不在同一直线以及间距也不完全相等，误差在锥桶半径范围内。如果车模未行驶入加油站直接通过每次罚 15 秒，从错误出口驶出不惩罚也不奖励。车辆每次成功通过加油站奖励减时间 10 秒。

## 2.2.4 坡道

基础赛道中设置有坡道，为了契合人工智能自动驾驶的技术路线，因此在坡



道前方 15 厘米处贴放坡道的标志，供人工智能模型的识别，便于提前预知坡道从而进行速控操作。

▲ 图 2.2.4 坡道

## 2.3 车模技术要求

### 2.3.1 车模平台

车模可以使用竞赛指定的 I 型车模。

车模必须带有车壳，保证车壳的完整美观。车壳必须完整的包裹车辆的本身，车身的底板外边缘，4 个轮子和越过车壳的传感器及其支撑件外，车辆的正视图，侧视图和俯视图看不到车辆的内部细节。为了方便调试和电池的安装，可以将车壳整体或部分做成活动的部件，但是在运行过程中必须保持车壳为闭合的整体。车壳的限制使用光敏树脂、PLA、ABS、尼龙，等塑料材质制作而成。

车模作品完成之后，车模的尺寸没有限制。

### 2.3.2 微控制器

车模的赛道元素检测识别需要只能使用百度 EdgeBoard 计算卡（FZ3B 赛事定制版）且只允许使用 1 块。

车模运动控制单片机使用 Infineon 单片机设计车模的运动控制电路板。

要求模型算法必须使用百度 Paddle 框架搭建，即必须使用百度深度学习框

架的人工智能算法实现。

### 2.3.3 传感器

车模作品中只允许最多使用 2 个摄像头对赛道进行识别，并且摄像头必须直连到 EdgeBoard 计算卡（可通过 HUB 拓展 USB 接口直连数量）上用于赛道及其元素的检测。

车模作品中允许使用其他非摄像头类传感器进行环境的辅助检测，车辆姿态和运动控制的反馈，但不得用于赛道元素（直道，弯道，坡道等）和赛道标志的识别。选用的传感器或者其它电子部件中不得包括独立的微处理器，超声波传感器除外。

## 第三章 智能车机械的结构调整与优化

本组别的智能车的硬件基础是在给定的 I 车模，需要在此基础上进行设计架构并搭建完成整车并完成比赛，因此在设计整车软件架构和算法之前需要结合整车机械结构有充足了解和认识，然后建立相应的数学模型并着手编写程序。从而再针对具体的比赛要求和规范来调整赛车的机械结构，并在实际的调试过程中不断的改进和提高。本章将主要介绍智能车车模的机械结构和调整方案。

### 3.1 智能车摄像头位置以及安装固定

在摄像头的选择上，我们采用了工业摄像头，选择了广角相对大的型号以便于尽量维持对赛道元素识别的范围，同时在摄像头的安装上对原车模进行修改，将摄像头的安装位置从原厂的电机上方修改为在电机与舵机之间，同时改变了高度以便抵消摄像头位置变化造成的结果。

### 3.2 车模内空间安排

由于本组别的特殊性——所有参赛车辆的内部电路不得裸露，需布置在车壳内部，故需要对全车各部分的安装位置进行精准安装，包括 EdgeBoard 板卡，下位机，电池，以及连接各部分的走线，按照一定顺序合理布局，以此避免导致电磁信号互相干扰以及走线缠绕。

### 3.3 自制车壳的设计和打印

在测试原装车壳后，发现经过我们改装后的车模与原装车壳的适配性较差，我们采用了自制车壳以便缓解较为复杂的自制车模内部环境对整车的不良影响。

### 3.4 前桥以及重心位置调整

现代汽车在正常行驶过程中，为了使汽车直线行驶稳定，转向轻便，转向后能自动回正，减少轮胎和转向系零件的磨损等，在转向轮、转向节和前轴之间须形成一定的相对安装位置，叫车轮定位，其主要的参数有：主销后倾、主销内倾、车轮外倾和前束。智能车竞赛模型车的四项参数都可以调整，但是由于模型车加工和制造精度的问题，在通用的规律中还存在着一些偶然性。车模最终要达到的效果就是重心稳定，可以达到人为很容易的就可以把小车放到一个自己可以平衡的位置，并且可以自己长时间处于平衡状态，这样说明小车的重心相对比较稳定了。重心前后方向的调整，对赛车行驶的性能有很大的影响，根据车辆运动学理论，车身重心前移，但会降低转向的灵敏度（因为大部分重量压在前轮，转向负载增大），同时降低后轮的抓地力；重心后移，但会增大转向的灵敏度，后轮的抓地力也会增加。因此，调整合适的重心，让车模更加适应比赛的赛道是非常关键的。电池的位置同时也决定车的速度。我们选择将电池安装在舵机左边，使得车模总量均匀，能够灵活转向。

车模总质量的探讨：我们大家都知道物体质量越大，它的惯性就越大，然而惯性越大对于智能小车来说就不容易停车，灵活性大大降低，这对于一个用程序控

制的小车是一大危害，所以在设计小车时，尽一切可能的减轻车模的整体质量，所以在画电路板时尽可能小。且谨慎给智能车添加配重。

## 第四章 硬件电路设计

### 4.1 硬件方案设计

硬件电路对于整个小车系统至关重要，只有好的硬件才能保证程序员在后期的调试效果，因此在做车前期要尽量设计出合理的电路。I 车模由于车模结构限制，安装硬件电路的。电磁信号放大电路对 PCB 的走线比较严格，把信号放大电路集成到主控板上放大信号容易被主控板上的电流干扰，最终决定把信号放大板子单独分割出来。一块是主控板和驱动板，我们把主控和驱动画在了一块板子上，我们的原则是在不影响电路的前提下，尽可能的把板子画的和车模后端相吻合，以便于我们后期的搭车。还有调参板，我们用的是把按键和 LCD 液晶屏结合的方式，这样可以合理地应用空间，保证了车的简洁。

### 4.2 供电管理

好的电源分配才能保证电路的正常工作。

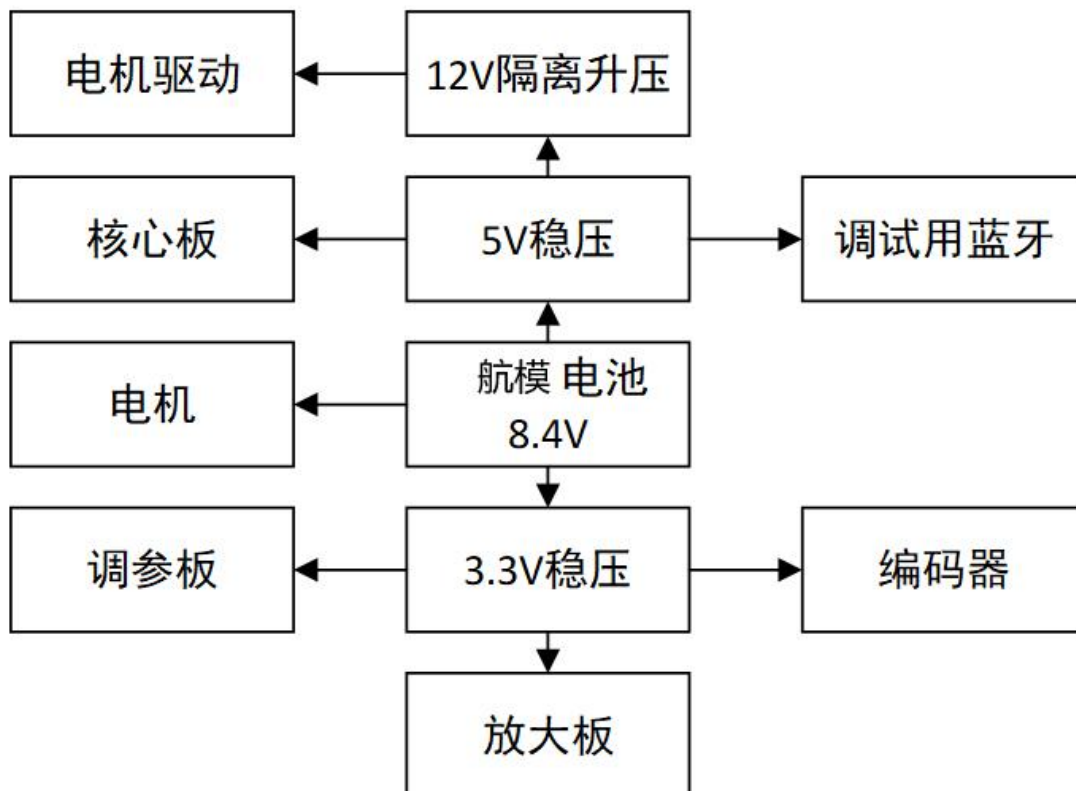
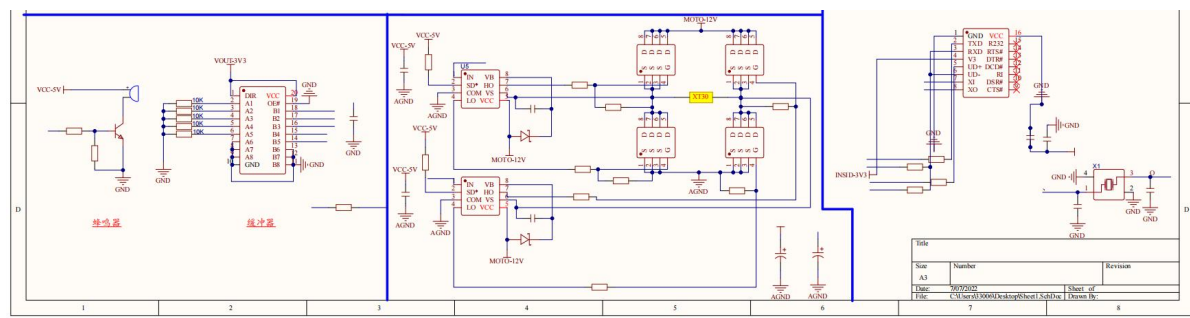


图 4-1 小车电源分配图

在参照前几年学长的经验，以及前几届的技术报告后，我们决定采用 TPS76833 和 TPS76850 进行稳压供电，这种芯片是低压差线性稳压类型的芯片，对电源纹波抑制较好，且稳定性好，可以一定程度上消除电机带来的电源波动影响，所以是一款不错的芯片，但是价格需要 20 元一片，在电源分配中，单片机及放大电路都是单独供电，编码器，LCD 调参板等都是 3.3V 供电。由于分立的 N 沟道 MOSFET 具有极低的导通电阻，大大减小了电枢回路总电阻。另外，专门设计的栅极驱动电路可以提高 MOSFET 的开关速度，使 PWM 控制方式的调制频率可以得到提高，从而减少电枢电流脉动。并且专用栅极驱动芯片通常具有防同臂导通、硬件死区、欠电压保护等功能，可以提高电路工作的可靠性。

### 4.3 主控电路和驱动电路

在考虑成本和稳定的情况下，我们使用 MOS 管搭建 H 桥，使用了国产 SE20N110 的 N 沟道 MOS 管，最大承受的瞬间电流达到 110A，对于驱动电机来说足够了，一片价格在 0.9 元左右，此款 MOS 管还需要 DRV8701 驱动芯片来驱动，一片价格也在 8 元左右，经过多次改良的电路，板子多余部分经过裁撤和重新设计，大幅精简。





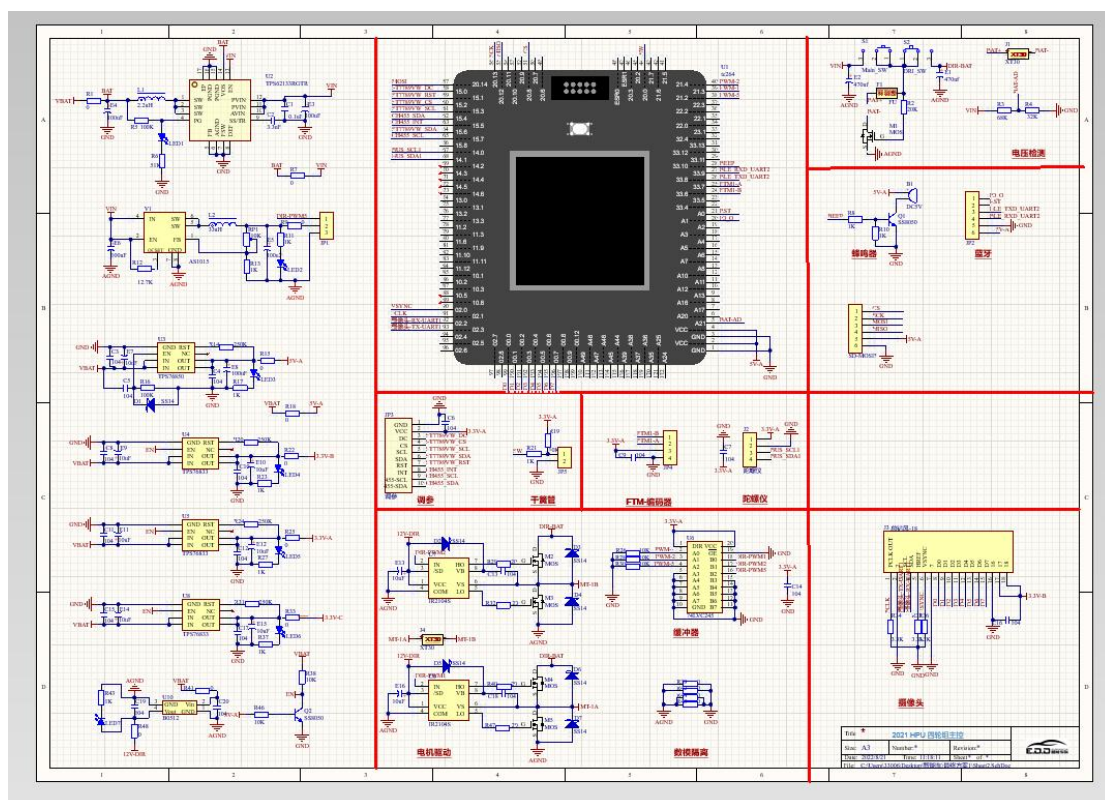


图 4-2 小车 PCB 原理图

## 4.4 总结

硬件制作完美是小车正常跑起来的基础，可以说是重中之重。一套好的硬件也是成绩提升的关键。小车的硬件确实很能影响整个队伍的成绩，硬件的稳定才能保证系统的正常工作，如果在调试时总是出现电路有问题的情况这样会很影响整个队伍的士气，因此在做车前期就应该设计较好的电路，在后期的经验积累下不断进步

## 第五章 智能车控制设计说明

高效的软件程序是智能车高速平稳运行的基础。我们设计的智能车系统采用 USB 彩色摄像头进行赛道识别，图像采集、模型训练、图像处理及校正处理就成了整个软件的核心内容。在智能车的转向和速度控制方面，我们使用了鲁棒性很好的经典 PID 控制算法，配合使用理论计算和实际参数补偿的办法，使智能车能够稳定快速寻线。

### 原始图像的特点

在 EdgeBoard 采集图像信号后需要对其分别进行模型分析和巡线分析处理以提取主要的赛道信息，同时，地标的存在、交叉道、起点线的存在，光线、杂点、赛道远处图像不清楚的干扰，图像效果会大打折扣。因此，在软件上必须排除干扰因素，对赛道进行有效识别，并提供尽可能多的赛道信息供决策使用。

在图像信号处理中我们提取的赛道信息主要包括：地标的特征以及位置、赛道两侧边沿点位置、通过校正计算的赛道中心位置，中心点规划面积，赛道变化幅度，赛道类型判别。

由于摄像头自身的特性，图像会产生梯形式变形，这使得摄像头看到的信息不真实。因此我们利用赛道进行测量，创建函数还原出了真实赛道信息。原始图像是一个将数字图像经模拟电路转换得到的二维数据矩阵，矩阵的每一个元素对应一个像素点，近处的图像大，黑线为梯形状。

模型程序将图像中训练好的特征地标进行识别预判断，需要时进行定位处理。

巡线程序上将每一行的黑白跳变点（跳变点按从右到左的顺序）记录下来，保存到两个二维数组里（分别表示上升沿、下降沿）。通过遍历上升沿和下降沿可以完成赛道边沿的提取。



## 5.1 赛道识别算法实现

### 5.1.1 图像预处理

通过 USB 摄像头传输回来的图像是 640\*480 大小的彩色图像，经过处理成灰度图像，而后调用自动阈值二值化后得到二值化图像，这种方法在图像处理的过程中仅仅有黑白两种元素，可以使图像处理更加简单。在实际的运行中当小车运行到不同的区域中的由于光线不均匀使得设定的单一阈值不同。通过之前的尝试，我们在仿真的过程中发现，使用最大协方差阈值法能够有效的计算出一幅图像的最佳阈值，实现对于图像前景与背景的分离。这样使用动态阈值二值化之后的图像能够像普通二值化图像一样处理，且基本能够适应不同区域的光线。模型部分图像修改大小为 320\*240 后投入进行检测。

### 5.1.2 基本扫线

智能车行驶是按照赛道中心线进行循迹，所以首要任务便是找出赛道的中心线。中心线可以由赛道左右边界加和除以 2 得到。对于左右边界的寻找，我们是先在图像底部 5 行按照从图像中间向两边寻找左右边线，遇到黑白跳变点即为赛道边界。找到底部的 5 行边界之后，根据赛道的连续性，下一个边界点以上一个边界点的左右 5 列进行寻找。这样可以节省扫线的时间。基本扫线如图 5-1 所示。

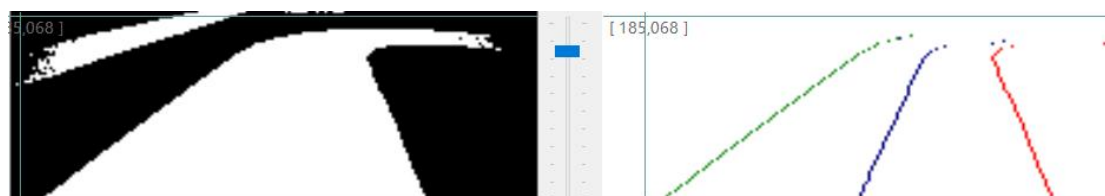


图 5-1-基本扫线

## 5.2 基础赛道元素识别

### 5.2.1 车库识别

车库是赛道中特征非常明显的元素，扫描一定区域内每一行的黑白跳变点个数，正常赛道的跳变点个数必定小于某一阈值，当超过阈值时就认为识别到了斑马线。

采用了固定打角及距离直接入库的方法，多次调试后即可精准入库。但此种方法容易受赛道摩擦系数等因素影响，稍有不慎就可能蹭到路肩，对控制方案也有较高要求。因此，对于入库，我们采取补线入库的方法。在判断到斑马线之后便可以寻找车库拐点进行拉线，补线操作。补线流程如图 5-2，图 5-3 所示。

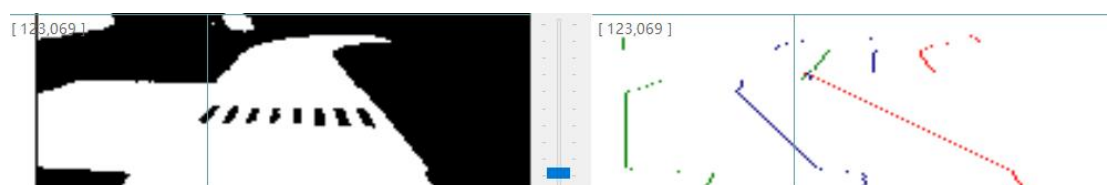


图 5-2-1 入库阶段 1



图 5-2-1 入库阶段 2

### 5.2.2 三叉识别

同时搜索左右两个三岔角点，通过判断连续几行的边界跳变来确定三岔的两个突变点。当搜索到两个三岔突变点后向上，计算出两个三岔角点的中点，并且根据这个角点向上搜索，当搜索到黑白跳变时即判断搜索到了三岔上角点，根据上角点以及三岔需要拐入方向的对立方向角点补线来成功通过三岔。



图 5-2-2 三叉识别补线图

### 5.2.3 十字识别

十字识别比较简单，十字路口处有四个非常明显的  $90^\circ$  角点，通过判断边界跳变以及斜率夹角可以很好地判断出下两个角点以及上两个角点。

- (1) 只搜索到下两个拐点时，根据下两个角点以下几行的边界斜率向上补线。
- (2) 只搜索到上两个拐点时，根据上两个角点以上几行的边界斜率向上补线。
- (3) 搜索到右侧下拐点和右侧上拐点时，通过两点确定直线法补出右边界。
- (4) 同时搜索到四个角点时，根据同侧上下两点连线可以补出完整边界。

### 5.2.4 圆环识别

对于圆环的识别与补线，我们采用了状态机补线思路，将一整个圆环分为（以左圆环为例）：

①状态一：进环前，左边有下拐点，右边为直线（判定方法为：最小二乘法拟合或者曲率计算）。



图 5-2-4 进环前

②状态二：进环中，此时找到左上拐点，链接左上与右下进行进环补线。



图 5-2-4 进环中

③状态三：在环中，此时正常循迹即可，当判断到出环特征时，进入状态四。

④状态四：出环，判断到右下拐点，即出环特征，此时进行出环补线操作，这时可以链接右下拐点与左上任意一点（根据自己的速度决定），进行出环补线。

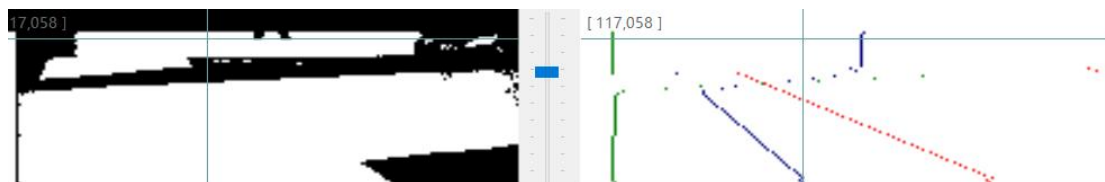


图 5-2-4 出环

⑤状态五：结束圆环：此时找到左上拐点，拟合左上拐点以上进行往下拉线，当左上拐点小于一定值或消失时，圆环状态结束。

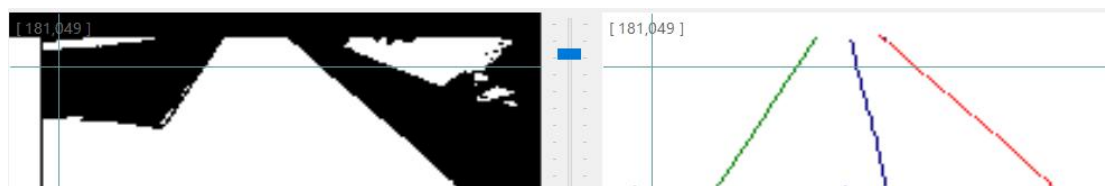


图 5-2-4 结束圆环

### 5.2.5 直道判断

我们对长直道的判断较为苛刻，即左右两条边线必须为从底部到顶部的连续长直线，并且顶部的赛道宽度小于一定值（用于区别远处的坡道和其他元素）。



图 5-2-5 直道

## 5.3 赛道特殊元素识别与定位

通过采集数据集投入训练，将训练好的模型调用 Paddlelite 进行模型部署工作，部署到 EdgeBoard 板卡上，读取调整过大小的图像进行地标检测与定位，识别后通过改变标志位进而在决策部分中进行特殊运动控制部分操作。

## 5.4 控制算法实现

PID (proportional-integral-derivative)控制是比例-积分-微分的简称。在生产过程自动控制的发展过程中，PID 控制是历史最悠久，生命力最强的基本控制方式。此后，随著科学技术的发展，特别是电子计算机的诞生和发展，涌现出许多先进控制策略，然而直到现在，PID 控制仍然得到广泛的应用，概括起来该算法具有原理简单，使用方便，适应性强，鲁棒性强等优点。

PID 调节器分为模拟式和控制式。前者采用运算放大器，阻容元器件等模拟电路构成，早起使用广泛，随著微处理器的发展，采用单片型微型计算机的数字式 PID 调节器引用越来越广泛。

### 5.4.1 模拟式 PID 调节器

以模拟量连续控制为基础的理想 PID 控制算法表达式如下：

$$u(t) = K_p \left[ e(t) + \frac{1}{T_i} \int_0^t e(t) dt + T_d \frac{de(t)}{dt} \right] \quad (\text{公式 1})$$

式中  $K_p$  是比例放大系数， $T_i$  是积分时间常数， $T_d$  是微分时间常数， $u(t)$ 是  $t$  时刻得输出， $e(t)$ 是  $t$  时刻控制器的输入， $e(t)=r(t) - y(t)$ ， $r(t)$ 和  $y(t)$ 分别是  $t$  时刻控制器的设定值和当前值。

在模拟 PID 控制器中，比例环节的作用是对偏差作出反应。偏差一旦产生控制器立即产生控制作用，使控制量向减小偏差的方向变化。控制作用的强弱取决于比例系数，比例系数越大，控制作用越强，则过渡过程越快，控制过程的静态偏差也

就越小；但是越大，也越容易产生振荡，破坏系统的稳定性。故而，比例系数选择必须恰当，才能过渡时间少，静态误差小而又稳定的效果。

积分环节的调节作用虽然会消除静态误差，但也会降低系统的响应速度，增加系统的超调量。积分常数越大，积分的积累作用越弱，这时系统在过渡时不会产生振荡；但是增大积分常数会减慢静态误差的消除过程，消除偏差所需的时间也较长，但可以减少超调量，提高系统的稳定性。当积分常数较小时，则积分的作用较强，这时系统过渡时间中有可能产生振荡，不过消除偏差所需的时间较短。所以必须根据实际控制的具体要求来确定积分常数。

微分环节的作用使阻止偏差的变化。它是根据偏差的变化趋势（变化速度）进行控制。偏差变化的越快，微分控制器的输出就越大，并能在偏差值变大之前进行修正。微分作用的引入，将有助于减小超调量，克服振荡，使系统趋于稳定，特别对高阶系统非常有利，它加快了系统的跟踪速度。但微分的作用对输入信号的噪声很敏感，对那些噪声较大的系统一般不用微分，或在微分起作用之前先对输入信号进行滤波。

### 5.4.2 数字式 PID 调节器

由于计算机控制技术的发展非常迅速，数字化 PID 算法得到了大量应用。由于计算机控制是一种采样控制，它只能根据采样时刻的偏差计算控制量，而不能像模拟控制那样连续输出控制量，进行连续控制。由于这一特点模拟 PID 公式中的积分项和微分项不能直接使用，必须进行离散化处理。离散化处理的方法为：以  $T$  作为采样周期， $k$  作为采样序号，则离散采样时间  $kT$  对应着连续时间  $t$ ，用矩形法数值积分近似代替积分，用一阶后向差分近似代替微分，可作如下近似变换：

$$\begin{cases} t \approx kT & (k=0, 1, 2, \dots) \\ \int_0^t e(t) dt \approx T \sum_{j=0}^{j=k} e(jT) = T \sum_{j=0}^{j=k} e_j \\ \frac{de(t)}{dt} \approx \frac{e(kT) - e[(k-1)T]}{T} = \frac{e_k - e_{k-1}}{T} \end{cases}$$

公式 2

上式中，为了表示的方便，将  $e(kT)$  简化成  $e(k)$ ，将上式带入模拟 PID 的公式中得到离散的 PID 表达式为：

$$u_k = K_p [e_k + \frac{T}{T_i} \sum_{j=0}^k e_j + T_d \frac{e_k - e_{k-1}}{T}] \quad \text{公式 3}$$

或：

$$u_k = K_p * e_k + T_i \sum_{j=0}^k e_j + T_d (e_k - e_{k-1}) \quad \text{公式 4}$$

其中  $k$ --采样序号， $k = 0, 1, 2 \dots$ ；

$u(k)$ --第  $k$  次采样时刻计算机的输出值；

$e(k)$ --第  $k$  次采样时刻偏差的输入值；

$e(k-1)$ --第  $k-1$  次采样时刻偏差的输入值；

$K_i$ --积分系数， $K_i = K_p * T / T_i$ ；

$K_d$ --微分系数， $K_d = K_p * T_d / T$ ；

如果采样周期足够小，则（公式 3）或（公式 4）的近似计算可以获得足够精确的结果，离散控制过程与连续过程十分接近。

（公式 3）或（公式 4）表示的控制算法直接按（公式 1）所给出的 PID 控制规律定义进行计算的，所以它给出了全部控制量的大小，因此被称为全量式或位置式 PID 控制算法。

这种算法的缺点是：由于全量输出，所以每次输出均与过去状态有关，计算时要对  $e(k)$  进行累加，工作量大；并且，因为计算机输出的  $u(k)$  对应的是执行机构的实际位置，如果计算机出现故障，输出的  $u(k)$  将大幅度变化，会引起执行机构的大幅度变化，有可能因此造成严重的生产事故，这在实际生产中是不允许的。

增量式 PID 控制算法可以避免这种现象发生。

所谓增量式 PID 是指数字控制器的输出只是控制量的增量  $\Delta u(k)$ 。当执行机构需要的控制量是增量，而不是位置量的绝对数值时，可以使用增量式 PID 控制算法进行控制。

增量式 PID 控制算法可以通过（公式 3）推导出。由（公式 3）可以得到控制器的第  $k-1$  个采样时刻的输出值为：

$$u_{k-1} = K_p [e_{k-1} + \frac{T}{T_i} \sum_{j=0}^{k-1} e_j + T_d \frac{e_{k-1} - e_{k-2}}{T}] \quad \text{公式 5}$$

将（公式 3）与（公式 5）相减并整理，就可以得到增量式 PID 控制算法公式为：

$$\Delta u_k = u_k - u_{k-1} = K_p [e_k - e_{k-1} + \frac{T}{T_i} e_k + T_d \frac{e_k - 2e_{k-1} + e_{k-2}}{T}] \quad \text{公式 6}$$

由（公式 5）可以看出，如果计算机控制系统采用恒定的采样周期  $T$ ，只要使用前后三次测量的偏差值，就可以由（公式 5）求出控制量。

增量式 PID 控制算法与位置式 PID 算法相比，计算量小的多，因此在实际中得到广泛的应用。

而位置式 PID 控制算法也可以通过增量式控制算法推出递推计算公式：

$$u_k = \Delta u_k + u_{k-1} \quad \text{公式 7}$$

（公式 7）就是目前在计算机控制中广泛应用的数字递推 PID 控制算法。

## 5.5 电机的控制

对于速度的控制，采用了编码器闭环的策略，编码器来反馈实际速度，小车根据实际速度与目标速度的差值，再通过 PID 算法的计算，最后根据计算得到的值修改



输出大小，以保证小车速度一直都是稳定的，这就是闭环的优点。

## 5.6 舵机的控制

舵机的控制就是根据处理图形之后返回的合适的偏差的进行一个打角，在偏移量的计算当中，我们通过对整副图中线加权的方式来计算偏移量。给每一行设定一个权值，最后通过所有有效行的加权来计算出当前图像的偏移量。PD 控制当中的偏差就是之前通过整幅图加权的方式得出。每一行加权的权重是经过长期整定的，最后得出了一套合理的，适应性强的权值。

## 5.7 速度的控制

通过对编码器的数据的滤波处理，获得车子的车子速度，并且使用了增量式 pid 与变积分融合控制，其优点在于在普通的 PID 控制算法中，由于积分系数  $K_i$  是常数，所以在整个控制过程中，积分增量是不变的。而真正车辆电机调试变速过程中在误差较大的时候，积分可能出现饱和状态，甚至超调，其实应该削弱积分或者消除。而误差较小的时候，应该增大积分以消除静差，可以让车的速度跟随大大提升。在增量式 PID 与变积分调试过程中，可以适当的超调 PID 以更大程度地做到速度跟随，从而弥补电流不稳定，造成车系统的不稳定。

## 第六章 系统开发与调试

### 6.1 下位机程序开发工具

下位机部分程序的开发是在下进行的 AURIX Development Studio，包括源程序的编写、编译和链接，并最终生成可执行文件。

AURIX™ Development Studio（以下简称 ADS）是英飞凌公司于 2019 年底推出的免费\*的集成开发环境，支持英飞凌 TriCore™内核 AURIX™ 系列 MCU；ADS 是一个完整的开发环境，包含了 Eclipse IDE、C 编译器、Multi-core 调试器、英飞凌底层驱动库（low-level driver,iLLD),同时对于编辑、编译及调试应用代码没有时间及代码大小的限制。

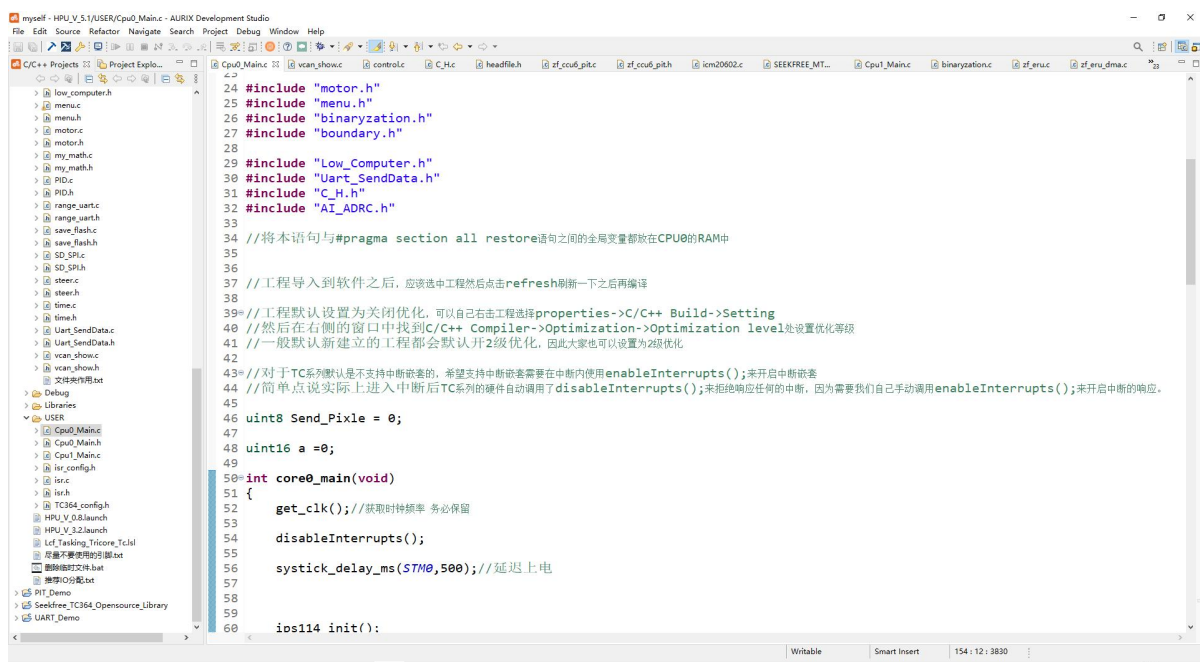


图 6- 1 编译画面

### 6.2 上位机开发工具

#### 6.2.1 上位机编写代码

车辆代码使用 vscode 远程 ssh 连接板卡进行代码编写工作，使用 winscp 进行文件传输工作，使用 VNC viewer 软件进行 vnc 远程界面显示上位机屏幕部分，另外还使用 Windows PowerShell 进行远程 ssh 连接快捷操作。

模型训练平台使用 AI Studio 在线平台，编写模型训练代码，投入采集的数据集进行训练模型，训练好通过 winscp 将模型文件传入车载 EdgeBoard 板卡之中。

## 6.2.2 下位机屏幕调参

每次在发车之前，对赛道信息采集完毕之后，都会通过调参板来设定其中的一些控制参数，比如转向环的 PID 值的大小，以及小车的速度，入库方式等。



图 6-2-2 调参屏界面

第七章 车模参数

车 模 技 术 检 查 表

队伍名称	HPU_OS			
参赛学校	河南理工大学			
赛题组组别	四轮电磁          四轮摄像头          多车编队 平衡单车          无线充电          平衡信标 智能视觉          极速越野 <input checked="" type="checkbox"/> 完全模型			
检查项目	规格 (选手自行填写)	符合 (√)	不符合 (×)	备注
1. 车模类型是什么？	I 车模	√		
车模整体尺寸： 1. （包括传感器在内）长，宽，高 (mm) 2. 对于无线充电组：显示电能的 LED 板尺寸	长： 292 宽： 178 高； 440	√		
1. 传感器种类、规格(型号)数量。	HF868-工业摄像头 数量： 1	√		
1. 控制转向舵机型号是否自行改装舵机？ 2. 防伪易损标签是否完整？	CS-3120(百度官方 I 车模电机) 标签完整	√		
1. 是否增加伺服电机？ 2. 如果有那么种类、个数和作用？	未增加伺服电机	√		
1. 微处理器型号和个	无			

数?		✓		
2. 是否复合所在比赛组别要求?				
1. 是否具有其它可编程器件, 个数与作用?	无	✓		
1. 是否有无线通讯装置?	USB 无线网卡 数量: 1	✓		
2. 如果有, 那么种类和个数?				
1. 电池的种类、规格和数量?	种类: 全国大学生智能车竞赛专用电池 规格: 2200mA 数量: 1	✓		
1. 是否有升压电路驱动舵机和后轮电机?	无	✓		
1. 后轮驱动电机是否是原车模电机?				
2. 是否具有防伪易损标签?	1. 采用原车模电机; 2. 具有防伪易损标签	✓		
1. 车模轮胎是否原有的纹理可辨析?	1. 车模轮胎是原有的纹理可辨析			C
2. 轮胎表面是否具有粘性物质?	2. 轮胎表面无粘性物质	✓		
3. 对于麦克纳姆轮是否更换过小轮胶皮?				
1. 车模底盘是否是原车模底盘?	1. 车模底盘是原车模底盘	✓		
2. 是否有大面积切割?	2. 无大面积切割			
1. 车轮轴距、轮距是否改装?	车轮轴距、轮距未改装	✓		
2. 改装参数是什么?				
1. 车模驱动轮传动机构	车模驱动轮传动机构未改装			

第十七届全国大学生智能汽车竞赛技术报告

是否改装？		✓		
2. 改装方式是什么？				
1. 车模差速器是否改装？	车模差速器未改装	✓		
2. 改装方式是什么？				
1. 车模零件是否更换或改装？	车模零件是未更换或改装	✓		。
2. 更换和改装的方式什么？				
1. 车模电路板个数及功能。	车模电路板个数：3	✓		
2. 其中是否有购买成品、哪一些？	功能：电机驱动；舵机驱动； 上下位机通讯，参数调整， ；电源转接 购买成品：USB 拓展坞			
1. 自制电路板是否标记有学校名称、队伍名称、制作日期等信息？	1. 自制电路板标记有学校名称、队伍名称、制作日期等信息；	✓		河南理工大学
2. 标示信息在PCB的哪一层？	2.信息标记在 pcb 开窗层、绿油层 (solder mask)			HPU_OS
				常炳星
				赵国浩
				席 文
				乔笑翊
				朱峻沅
其它待说明内容				
检查人员签名：	检查意见：			

## 第八章 参赛心得体会

站在国赛前回首过去的调车一年的经历，感慨万千。从 2021 年十一月十六号组建全模型队伍，我与赵国浩、席文、乔笑翊、朱峻沅四位同学便开启曲曲折折的探索学习之路，从十一月份到二月份的分工学习新知识新领域以及在废车模上搭建基础四轮小车进行巡线补线学习的渐入佳境，到三月四月份着手搭建完全模型车模、电路及线上赛调试的热火朝天，继而就进入黑暗的五月份小车落地之后舵机、EdgeBoard 板卡相继烧坏，疫情影响快递不通，转用其他组舵机和树莓派搭建小车进行调试，之后进入艰苦奋斗的六月七月，提前放假加不能留校，大家只好集中到农村调车，也就是自此才开始正式没日没夜调车，这些日子里大家给我很多感动很多力量，我觉得我们团队能够推进智能车到华北赛区第四的名次缺一不可，付出大量精力，作为队长我真的很感谢大家能够相互陪伴走完这一遭。

短短几个月，起起伏伏之中我们团队有若干次方案尝试、若干次赛道搭建、若干次发车、若干次冲出赛道、若干次完成比赛，每次完赛都是那么艰辛与激动。

智能车比赛在我看来并非简单的对电子电路和软件代码进行研究学习，而是从团队沟通、分工协作到相互用行动陪伴、鼓励彼此，一如既往地推进项目的进行，不论遇到什么坎坷，都积极寻找替代解决方法以及积极准备预案。经历这一遭智能车改变了我许许多多的固有思维，心态方面也得到锻炼，与小伙伴相处不能说很和睦，但在推进项目方面是一以贯之的，我觉得智能车参赛经历我受益终生。

再见了智能车。

最后仍要感谢所有为继承学长队名 HPU\_OS 呕心沥血的队友们，感谢一路以来为我们指导的学长们和老师，感谢这一年中所有给予我们帮助的可爱的人们。

谢谢你们。

常炳星

2022/8/20

## 参考文献

- [1]. 斋藤康毅著, 陆宇杰译,《深度学习入门-基于 python 的理论与实践》.人民邮电出版社, 2018.
- [2]. [美] 伊恩·古德费洛 / [加] 约书亚·本吉奥 / [加] 亚伦·库维尔,《深度学习》.人民邮电出版社, 2017.
- [3]. Alex Galea/Luis Capelo,《用 python 进行深度学习》.
- [4]. (加)亚历克斯·盖利(Alex Galea)、(古)路易斯·卡佩罗(Luis Capelo)、高凯、吴林芳、李娇娥、朱玉,《Python 深度学习应用》.清华大学出版社, 2020.
- [5]. 程天恒,《Paddle Paddle 与深度学习应用实战》.电子工业出版社, 2018.
- [6]. 基于卷积神经网络的智能车前方车辆检测系统研究\_曹聪聪
- [7]. <https://zhuanlan.zhihu.com/p/33667203> 卷积神经网络 CNN 原理详解(一)——基本原理
- [8]. <https://baike.baidu.com/item/%E6%B7%B1%E5%BA%A6%E5%AD%A6%E4%B9%A0/3729729?fr=aladdin>——百度百科——深度学习
- [9]. <https://www.w3cschool.cn/opencv> --opencv 官方文档



## 附 录

比赛部分源代码

```
if(sgq_progress!=2&&sgq_progress!=3&&ImageStatus.Road_type != Forkin){  
    if(ImageStatus.Road_type != Barn_in)  
        drawlines();  
    else  
        Filter_Find_Seed_01(54,59);  
    draw_lines_all();    //处理其它行的  
边 10us  
    image_basic();  
    yuansushibie(); //赛道元素识别  
    //1ms  
    DrawExtensionLine();           //绘制  
制延长线并重新确定中线，把补线补成斜线 100us  
    RouteFilter();    //70us  
}  
if(ImageStatus.Road_type == Forkin){  
    ImageStatus.TowPoint = 24;  
    if(Fork_progress == 3)  
        ImageStatus.TowPoint = 24;
```

```
        if(Fork_progress == 4)
            ImageStatus.TowPoint = 19;
    }
    if(ImageStatus.Road_type == shigongqu)
    {
        if(sgq_progress == 1)
            ImageStatus.TowPoint = 25;
        if(sgq_progress == 3)
            ImageStatus.TowPoint = 45;
    }
    if(ImageStatus.Road_type ==
LeftCirque||ImageStatus.Road_type
==RightCirque)
    {
        if(Circle_progress==3){
            ImageStatus.TowPoint = 33;
        }
        if(Circle_progress==4){
            ImageStatus.TowPoint = 35;
            printf("center:%d",ImageDeal[2
9].Center);
        }
    }
```

```
        if(Circle_progress==5)
            ImageStatus.TowPoint = 34;
        if(Circle_progress==6)
            ImageStatus.TowPoint = 19;
    }
    if(ImageStatus.Road_type == Barn_in &&
ImageStatus.Barn_Flag == 2)
        ImageStatus.TowPoint = 37;
    if(ImageStatus.Road_type ==Cross)
        ImageStatus.TowPoint = 30;
    // sgq_test();
    printf("sgq_flag1:%d\n",sgq_flag1);
    printf("sgq_flag2:%d\n",SGQ_flag2);
    sgq_Handle();
    Cross_Handle();
    sancha_driver();
    Cirque_driver();
    Barn_in_Handle();
    Barn_out_Handle();
    GetDet();           //7us
```