# 十一届“恩智浦”杯全国大学生智能汽车竞赛

# 技 术 报 告


学　　校：　南昌航空大学

队伍名称：　摩卡飞鹰

参赛队员：　莫嘉明、邹海英、蒋小文

带队教师：　陈黎娟、吴开志

# 关于技术报告和研究论文使用授权的说明

本人完全了解第十一届“恩智浦”杯全国大学生智能汽车竞赛关保留、使用技术报告和研究论文的规定，即：参赛作品著作权归参赛者本人，比赛组委会和恩智浦半导体公司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：莫嘉明 蒋小文 邹海英

带队教师签名：[illegible] 吴开发

日　　期：8月14号，2016年

# 摘要

本文设计的智能车系统以MK60DN512ZVLQ10微控制器为核心控制单元，基于CMOS摄像头的图像采样获取赛道图像信息，提取赛道中心线，计算出小车与黑线间的位置偏差，采用PD方式对舵机转向进行反馈控制。使用PID控制算法调节驱动电机的转速，结合特定算法分析出前方赛道信息实现对模型车运动速度的闭环控制。为了提高调试参数的效率，采用Java开发了蓝牙调试的APP，以及液晶显示模块。

关键词：MK60DN512ZVLQ10，CMOS摄像头，PID控制，蓝牙

# Abstract

In this paper, design of the intelligent vehicle system with MK60DN512ZVLQ10 micro controller as the core control unit, based on CMOS camera image sampling circuit of image information, to extract the centerline of the track calculated between the car and black line position deviation, of the steering gear steering by way of PD feedback control.Using PID control algorithm to adjust the speed of the drive motor, combined with the specific algorithm analysis information out of the track ahead of model car speed closed-loop control.In order to improve the efficiency of the debug parameter, we also use Java development bluetooth to debug the APP, and liquid crystal display module.

**Key words**: MK60DN512ZVLQ10, CMOS camera, binarization, PID control, the bluetooth

# 目录

# 引言

在半导体技术日渐发展的今天，电子技术在汽车中的应用越来广泛，汽车智能化已成为行业发展的必然趋势。汽车智能化被认为是汽车技术发展进程中的一次革命，汽车智能化的程度被看作是衡量现代汽车水平的重要标志，是用来开发新车型，改进汽车性能最重要的技术措施。有研究认为智能汽车作为一种全新的汽车概念和汽车产品，在不久的将来会成为汽车生产和汽车市场的主流产品。

全国大学生“恩智浦”（飞思卡尔）杯智能汽车竞赛从 2006 年开始，至今已经举办到了第十一届。该赛事与教育部已经举办的四大竞赛一样，都是为了提高大学生的动手能力和创新能力而举办的，具有重大的现实意义。与其他大赛不同的是，这个大赛的综合性很强，是以迅猛发展的汽车电子为背景，涵盖了控制、模式识别、传感、电子、电气、计算机和机械等多个学科交叉的科技创意性比赛，这对进一步培养本科生获取知识、应用知识的能力及创新意识，培养硕士生从事科学、技术研究能力，培养博士生知识、技术创新能力具有重要意义。

本技术报告主要包括机械结构、硬件系统、软件系统等内容，清晰的阐述了我们在智能车制作过程中所遇到的问题和所运用的解决方案。这份报告凝聚了我们的智慧，是我们团队共同努力的成果。

# 第一章 系统总体设计

## 1.1 系统分析

该系统以 32 位单片机 MK60DN512ZVLQ10 作为系统控制处理器， 采用基于的摄像头的图像采样模块获取赛道图像信息，以 PD 方式对舵机转向进行反馈控制，电机转速控制采用 PID 控制，通过 PWM 控制驱动电路调整电机的功率，通过特定算法分析出前方的路况，并根据路况的不同为小车选取最优路径在最短的时间内到达终点。

智能车系统主要包括以下模块：K60 最小系统、转向舵机模块、电机驱动模块、编码器、视频信号处理模块、蓝牙调试模块。

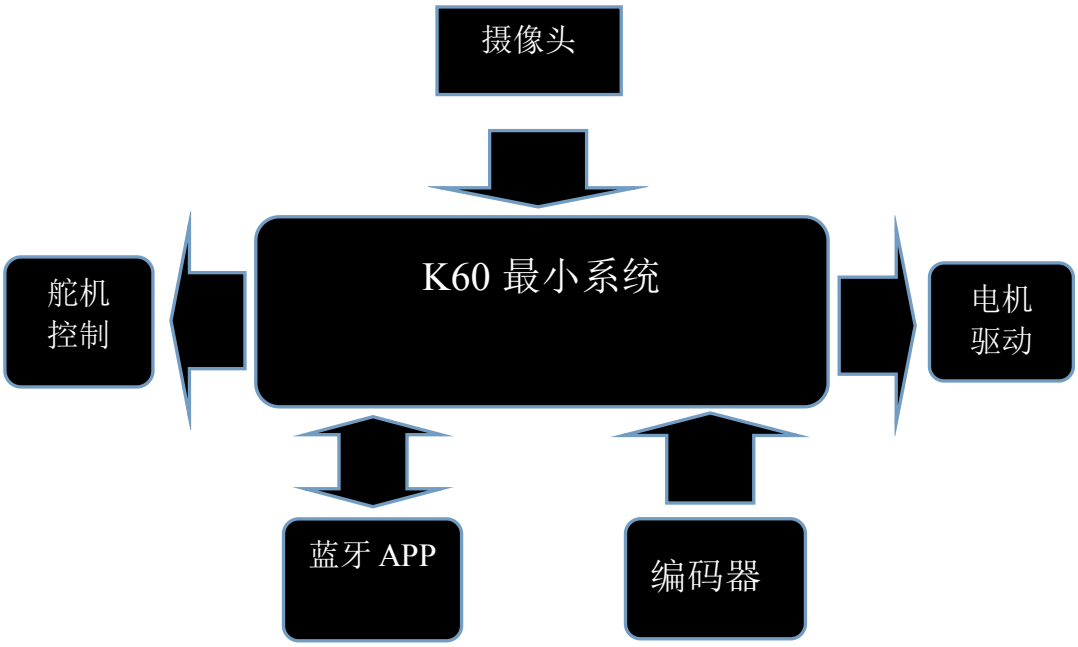



图 1.1 整体结构框图

## 1.2 车模整体布局

（1）为保证整车质量尽量小，电路元件多为贴片，PCB 尽可能的画小，支架及

固定装置用料少，材质轻。

（2）为降低赛车重心，底盘放至最低，主板电池低位放置；

（3）舵机直立安装，以提高舵机响应速度；

（4）采用高强度轻质量的碳素杆做摄像头支架；

（5）摄像头架在放在整车稍靠前位置

小车整体布局如下图所示：

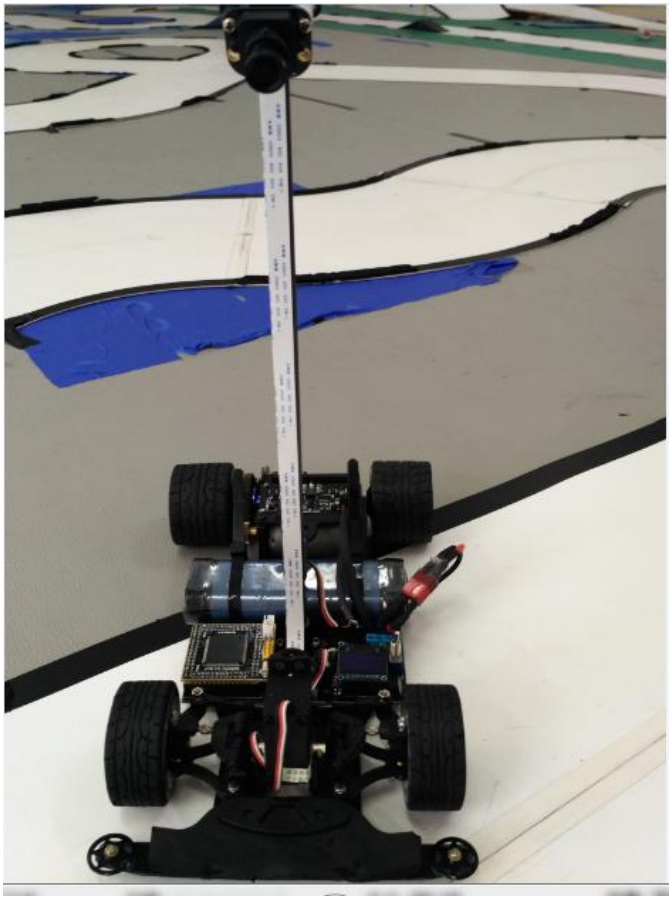


图 1.2 智能车正视图

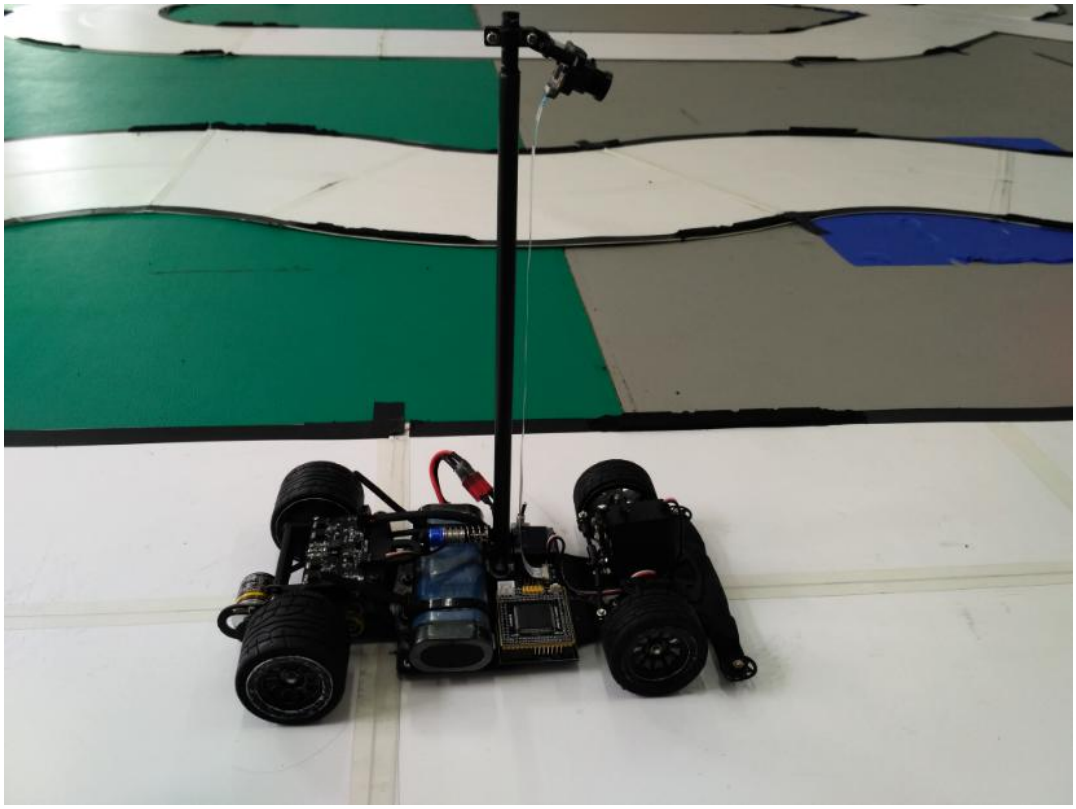


图 1.3 智能车侧视图

## 1.3 本章总结

智能车能够稳定的运行，良好的机械性能是必不可少的，机械安装的布局、质量将影响到道路信息的探测效果以及车子的行驶拐弯、加减速效果。只有整体把握好之后，才能对细节进行有效处理。本章主要介绍了智能小车控制系统的工作原理和赛车整体结构设计框图，使得赛车具有较好机械特性。

# 第二章 系统机械设计及实现

在整个调试过程中我们发现，车辆在高速情况对整车机械性能要求很高，前期由于经验不足，没有考虑的机械的因素，车模运行速度很容易遇到瓶颈，难以提高，为了能够使车在高速情况下更稳定流畅地运行，我们在后期装配时，重新对车的机械进行了整改和调。根据比赛规则要求，第十一届摄像头组采用 B 车模。固以下通过前轮 定位、舵机安装、后轮驱动与差速调节、传感器安装、车模加固及重心降低五个 方面就对 B 车模的改造进行介绍。

## 2.1 前轮定位

前轮定位对赛车的速度有很大的影响，虽然车模比较小，但是前轮定位作用还是不容忽视的。前轮定位的内容有：主销后倾、主销内倾、前轮前束。

### 2.1.1 主销后倾

主销后倾（见图 3.2）是指前轮主销在车模的纵向平面内（小车的侧面） 有一个向后的倾角 A，即主销轴线与地面垂直线在车模在纵向平面的夹角，称 为“主销后倾角”。采用主销后倾，车模在车轮偏转后会产生一回正力矩，纠 正车轮偏转。B 车模主销后倾是固定的，不能调节。

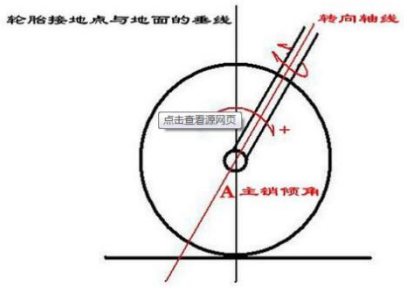



图 2.1 主销后倾角示意图

### 2.1.2 主销内倾

前轮主销在车模的横向平面内向内倾斜一个夹角，即主销轴线与地面垂直线 在车模的横向断面内的夹角，称为“主销内倾角”。

主销内倾角也有使轮胎自动回正的作用，当汽车转向轮在外力作用下发生偏转时，由于主销内倾，则车轮连同整个汽车的前部将被抬起一定高度，在外力消失后，车轮就会在重力作用下力图恢复到原来的中间位置。同时主销内倾角不宜过大，否则在转弯时轮胎将与赛道间产生较大的滑动，从而会增加轮胎与路面间的摩擦阻力，这不仅会使转向变得沉重，而且加速轮胎的磨损。因此，一般主销内倾角β不大于 8°。经过多次的试验，最终主销内倾角大约为 4°。

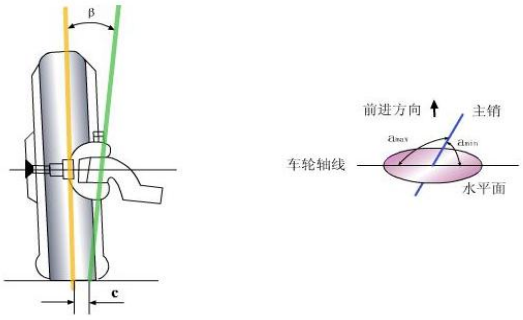



图 2.2 主销内倾角示意图

### 2.1.3 前轮前束

前轮前束是指前轮前端面与后端面在汽车横向方向的距离差，也可指车身前进方向与前轮平面之间的夹角，此时也称前束角。为保证汽车稳定的直线行驶，应使转向轮具有自动回正作用，即当转向轮在偶然遇到外力作用发生偏转时，在外力消失后能立即自动回到直线行驶的位置。这种回正作用是由转向轮的定位参数来保证实现的。适当选择前束角，可使前束引起的侧向力与车轮外倾引起的侧倾推力相互抵消.从而避免了额外的轮胎磨耗和动力的消耗。因此通常可以说，前束角是因外倾角的需要存在的。前束影响直线行驶和转向行驶，对于前轮驱动的车辆，它可以补偿运动轨迹上合成弹性运动的变化。对于标准驱动形式的车辆，前束约为 5′～20′。对于前轮驱动者（FDW），前束可达 20′(为补偿驱动力)。

最终给的前束较小，大约为 5′ 。

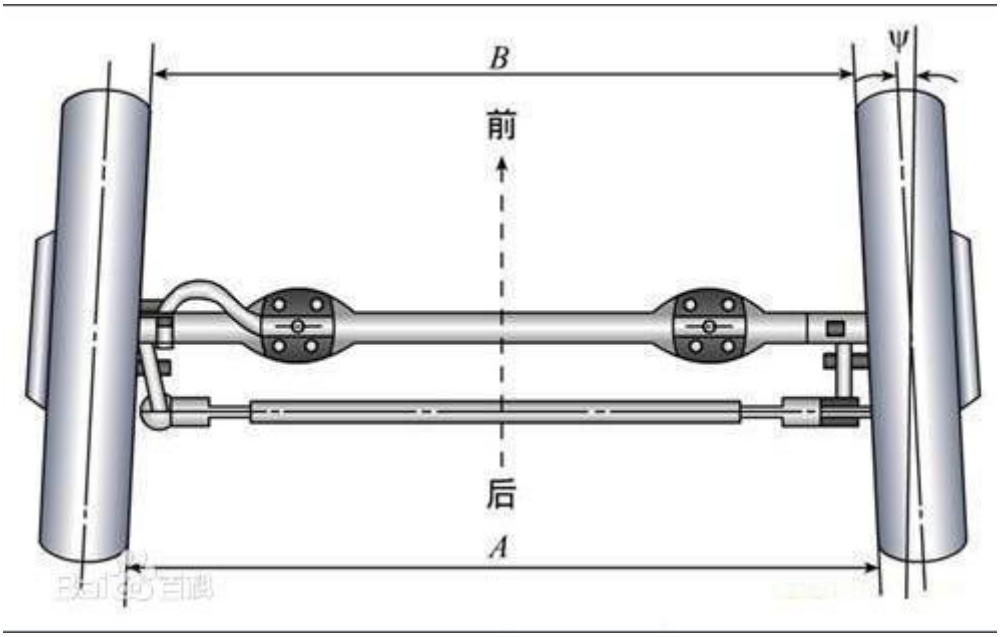



图 2.3 前束示意图

## 2.2 舵机安装

对于舵机的安装并没有经过严谨的计算，因为实验室缺少足够的更换材料，只是通过现有的器材达到一个目前最优的效果。使用合适的舵机轴线和转向连杆，使舵机连杆尽量平行。最终的舵机摆杆在 45mm 左右，拥有较大的打脚，适合转弯，虽然摆杆过长造成了一定灵敏度有一定的下降，但是我们在程序方面已经弥补了这一缺陷。并且，舵机安装要尽量向阿克曼打角原理靠拢：依据阿克曼转向几何设计的车辆，沿着弯道转弯时，利用四连杆的相等曲柄使内侧轮的转向角比外侧轮大大约 2~4 度，使四个轮子路径的圆心大致上交会于后轴的延长线上瞬时转向中心，这样可以使车辆在过弯时转向轮处于纯滚动状态，减少过弯时的阻力，减小轮胎的磨损。

理想效果如图：

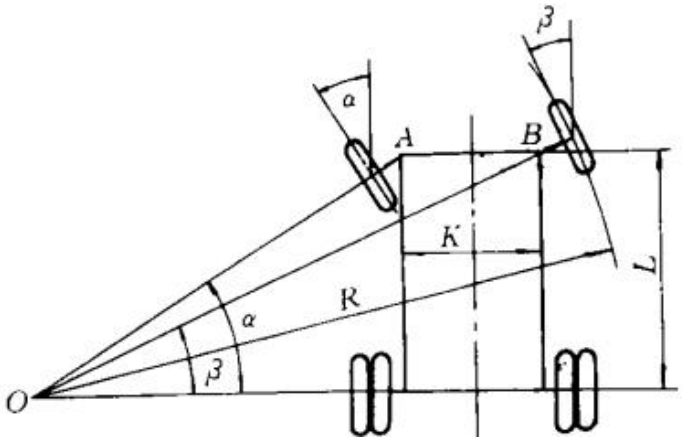



图 2.4 阿克曼转向原理示意图

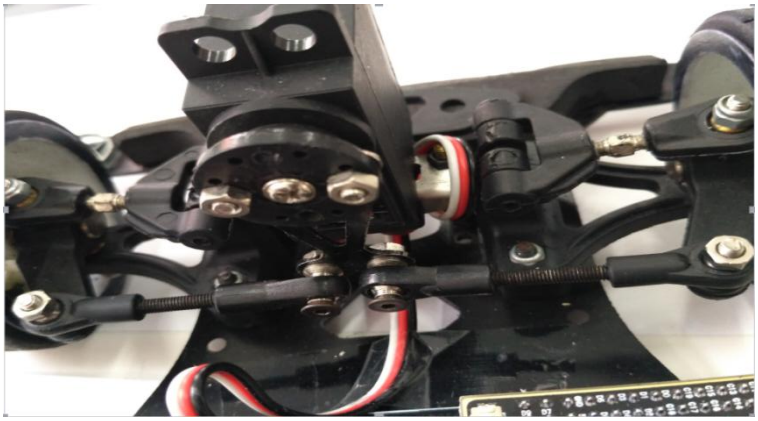


图 2.5 舵机安装最终效果图

### 2.2.1 左右转弯不对称问题

在调试的过程中，小车遇到了一个棘手的问题，很长时间都难以有效解决，经常是右转困难，左转容易。这个问题时后期经过机械整改后才暴露出来的。为此，我们对小车的机械重新进行了分析，最终问题定位在底盘不够平整，由于底盘的不平整，导致左右轮与赛道接触面积存在较大差异，左右转不对称也就很好解释了；问题虽然找到了，但是却难以找到一个合适的方法去解决，最终选择通过机械方式矫正底盘，时间久了，底盘也渐渐恢复了平整。综合对后轮差速已经前轮倾角的调整，终于，小车的左右转基本一致了。

## 2.3 编码器的安装

编码器是智能小车速度反馈元件，其安装位置应该充分考虑测速的准确性和稳定性，因此选用的是 500 线的欧姆龙编码器，可以达到很高的精度，符合我们的要求。在安装编码器的时候要保证有合适的齿轮咬合。咬合完美的原则是：两个传动齿轮轴保持平行，齿轮间的配合间隙要合适，过松容易打坏齿轮，过紧又会增加传动阻力；传动部分要轻松、顺畅，容易转动。判断齿轮传动是否调整好的一个依据是，听一下电机带动后轮空转时的声音。声音刺耳响亮，说明齿轮间的配合间隙过大，传动中有撞齿现象；声音闷而且有迟滞，则说明齿轮间的配合间隙过小，咬合过紧，或者两齿轮轴不平行，电机负载加大。调整好的齿轮传动噪音小，并且不会有碰撞类的杂音。


图 2.6 编码器安装示意图

## 2.4 摄像头的固定

为了确保摄像头能稳定循迹而不晃动，摄像头的固定十分重要。一是图像的失真要小，二是是整车重心要低，三是前瞻要比较大。综合考虑选用了合金底座，增强摄像头主杆的稳定性，支撑杆选用碳纤维管，摄像头高度为 30cm，既兼顾到了前瞻又不至于太高而抬高重心。并且，为了摄像头足够稳定，选用了 10mm 外径的碳纤维管，内嵌一根 6mm 粗的细管，便于摄像头的固定。

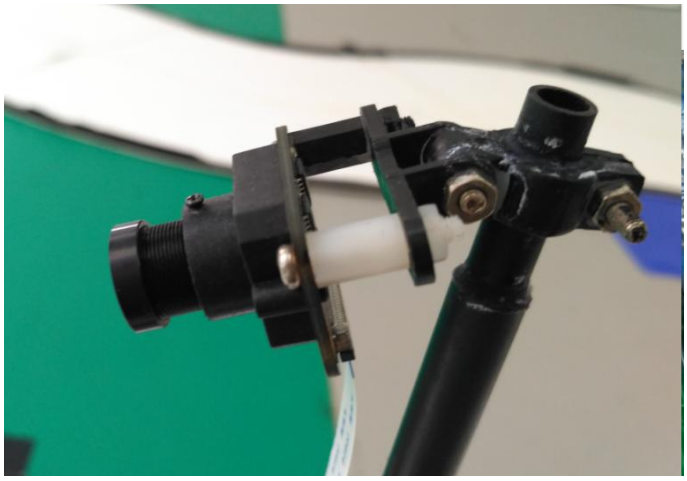


图 2.7 摄像头安装

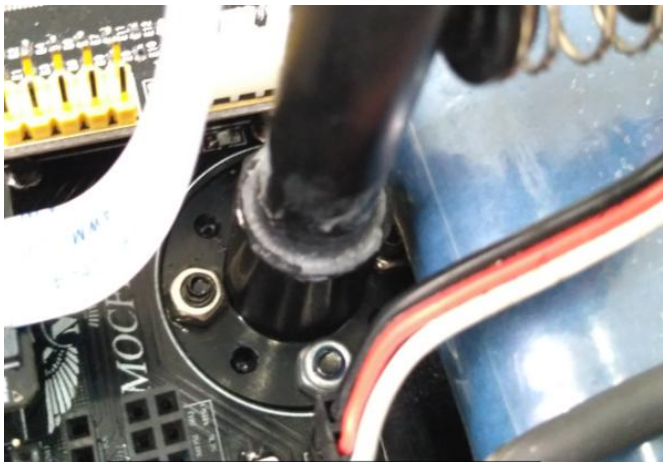


图 2.8 摄像头底座安装

## 2.5 车模底盘的高度及固定

车模底盘的高度主要由赛道中的坡决定，在顺利过坡的前提下底盘越低越好。这样会使得重心更低，赛车的稳定型更强。B 车模的底盘高度可调，我们选择较低的方案，并且能够顺利过坡道。底盘固定采用硬连接，但并没有用过多板子加固，在车模电池上方加有弹簧，避免小车因为中间过重产生无法看见的形变，并且能有效防止轻微的震动。车模前端加有防撞片，防止车模运行过程中冲出赛道造成损伤，防撞片上加有四驱车导轮属于个人喜爱，与机械关系不大。

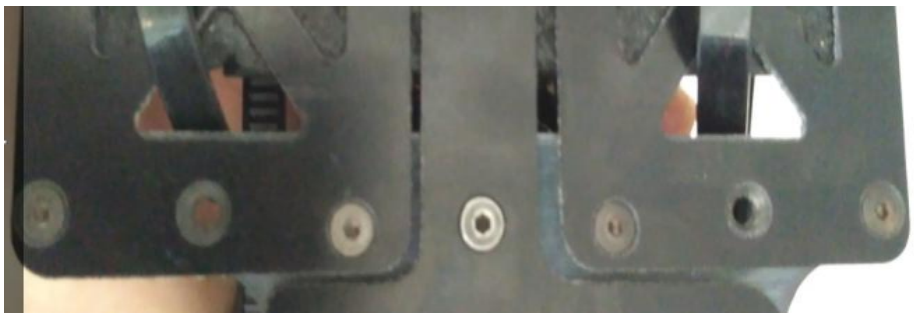


图 2.9 车模底盘连接

图 2.10 车头安装

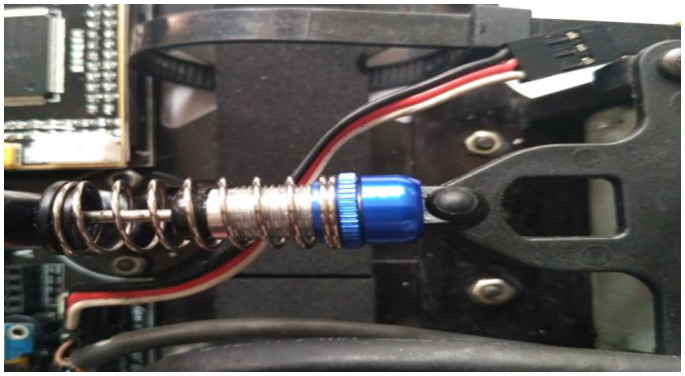


图 2.11 车模的固定

## 2.6 本章小结

本章主要介绍了车模的安装制作过程中几个重要的方面，包括前轮，舵机，摄像头，底盘以及编码器。一开始，我们还没有察觉出机械的重要性，越到后期越觉得机械结构的重要性。机械安装时一个需要全盘考虑的问题，机械安装的布局、质量将影响到道路信息的检测效果和小车的行驶路径。并且算法跟机械结构需要同时优化，才能把两者的优势发挥出来。

# 第三章 硬件电路设计

本章将重点介绍智能车系统硬件的设计与调试，该部分又分为 MK60DN512 单片机，电机驱动模块，速度检测模块，电源管理模块，信号采集模块，辅助调试模块。

## 3.1 硬件设计方案

可靠、高效、简洁是我们硬件设计的原则，选用性能优秀而且性价比好的芯片，设计稳定可靠兼容性好的电路板。

靠性是系统设计的第一要求，在电路设计的所有环节都进行了电磁兼容性设计，做好各部分的接地、屏蔽、滤波等工作，将高速数字电路与模拟电路分开，使本系统工作的可靠性达到了设计要求。

高效是指本系统的性能要足够强劲，使用了由分立元件制作的直流电动机可逆双极型桥式驱动器，该驱动器的额定工作电流可以轻易达到几十安培以上，保证了电动机的工作转矩和转速。

简洁是指在满足了可靠、高效的要求后，为了尽量减轻整车重量，降低模型车的重心位置，应使电路设计尽量简洁，尽量减少元器件使用数量，缩小电路板面积，使电路部分重量轻，易于安装。下图为硬件主要包含的模块。

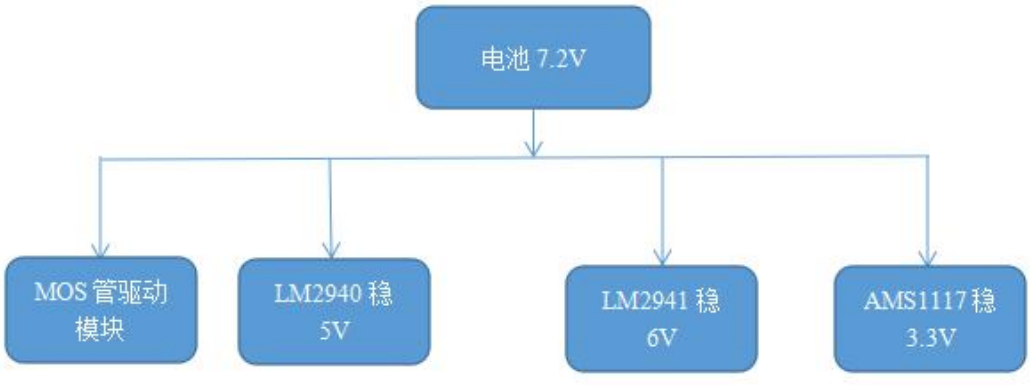



图 3.1 电源管理

## 3.2 核心板模块

在智能车的调试过程中，使用自行设计并制作的核心板。由于纯手工焊接的板子稳定性不足，因此，比赛时采用的购买的的最小系统板。芯片型号为K60DN512ZVLQ10。

K60DN512ZVLQ10 是 Kinetis 系列单片机中的一款增强型 32 位单片机，片内资源丰富，接口模块包括 SPI、SCI、IIC、A/D、FTM 等，在汽车电子应用领域具有广泛的用途。该微控制器系列具有以下性能：IEEE 1588 以太网，全速和高速 USB 2.0 On-The-Go 带设备充电探测， 硬件加密和防窜改探测能力。 丰富的模拟、 通信、 定时和控制外设从 100 LQFP 封装 256 KB 闪存开始可扩展到 256 MAPBGA 1MB 闪存。 大闪存的 K60 系列器件还可提供可选的单精度浮点单元、NAND 闪存控制器和 DRAM 控制器。

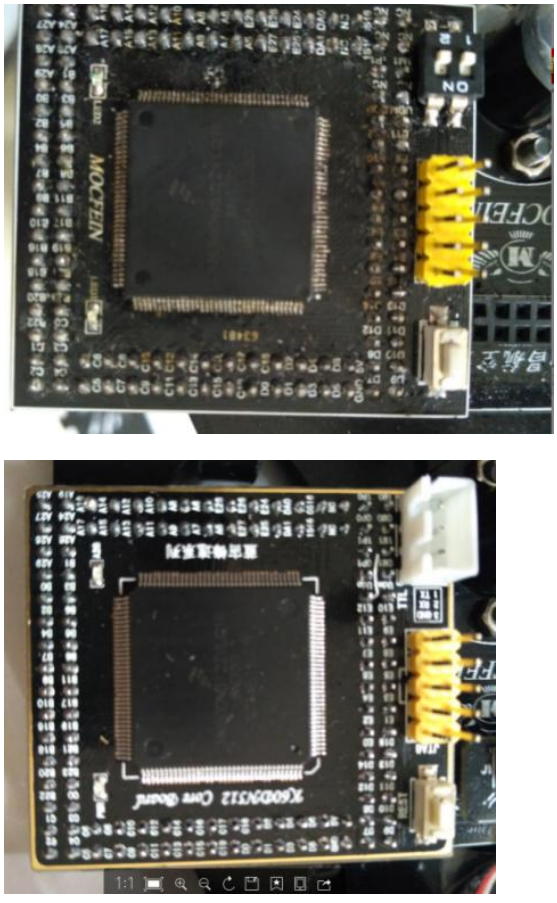


图 3.2 自制的核心板（左）购买的核心板（右）

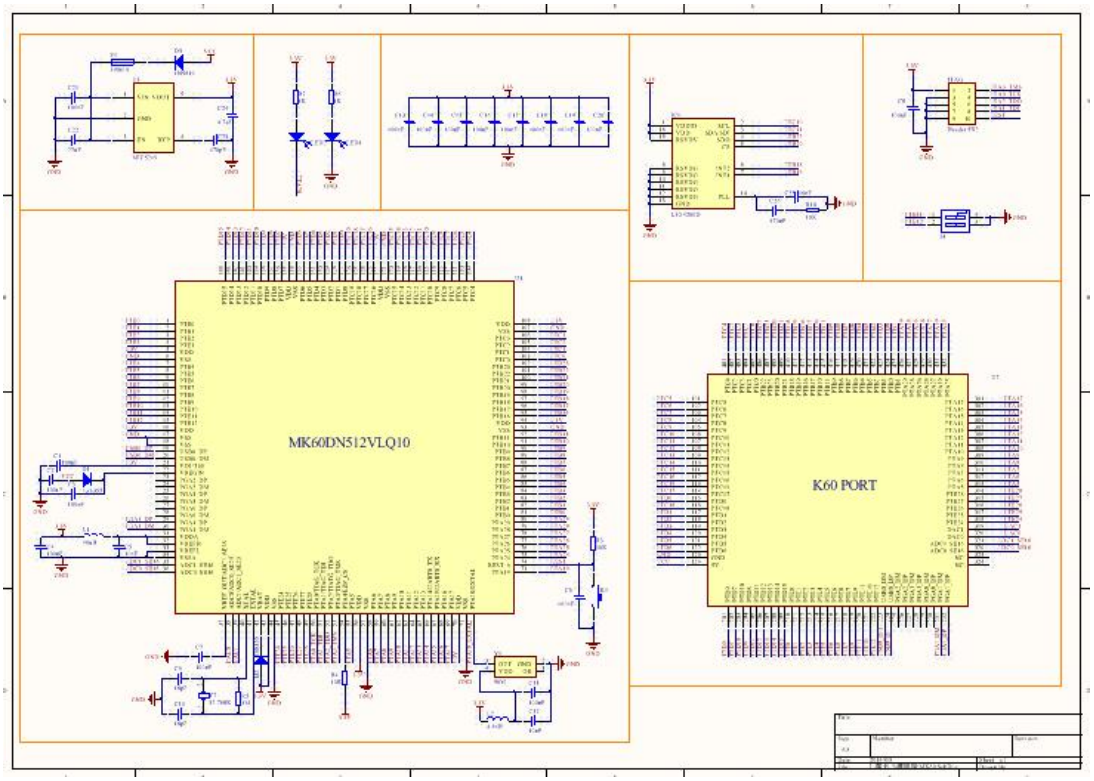



图 3.3　K60 核心板原理图

## 3.3 电源管理模块

电源管理是指如何将电源有效分配给系统的不同组件，电源管理对于依赖电池电源的智能小车是至关重要的。好的电源管理不仅能延长电池的使用寿命，更重要的是保证系统中各模块供电的稳定性，关系到整个系统能否正常工作。

电源调节器件通常使用最多的是线性稳压器件　如 78xx 系列三端稳压器件。虽然这种线性稳压器具有输出电压恒定或可调、稳压精度高的优点。但是由于其线性调整工作方式在工作中会造成较大的“热损失”　其值为压降(V)×负荷(I)，因此一般会导致其电源利用率不高、工作效率低下，不易达到便携式设备对低功耗的要求。 整车的硬件电路电源由可充电镍镉电池提供（7.2V、2000mAh)。但是系统中的各个电路模块所需要的工作电压和工作电流各不相同，所以设计了多种稳压电路，将电池电压转换成各个模块所需要的电压。

电压种类大致有 3.3V 为蜂鸣器、OLED 供电；5V 为摄像头、编码器和蓝牙

电路供电；4V-6V 为舵机提供工作电压；12V 为驱动芯片供电；其中 3.3VAMS1117 获得；12V 是由芯片 MC34063 升压后提供的；5V 是通过 LM2940 稳压稳压后获得，4V-6V 是通过 LM2941 变压后获得。

### 3.3.1 舵机供电模块

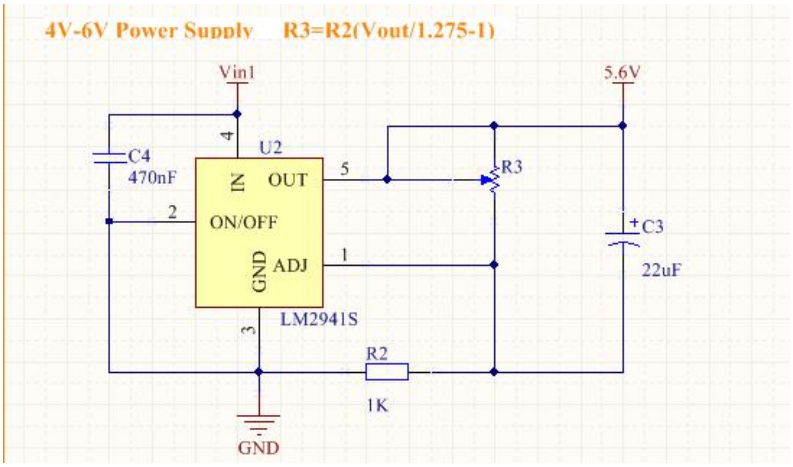



图 3.4 舵机供电原理图

### 3.3.2 蜂鸣器、OLED 供电模块

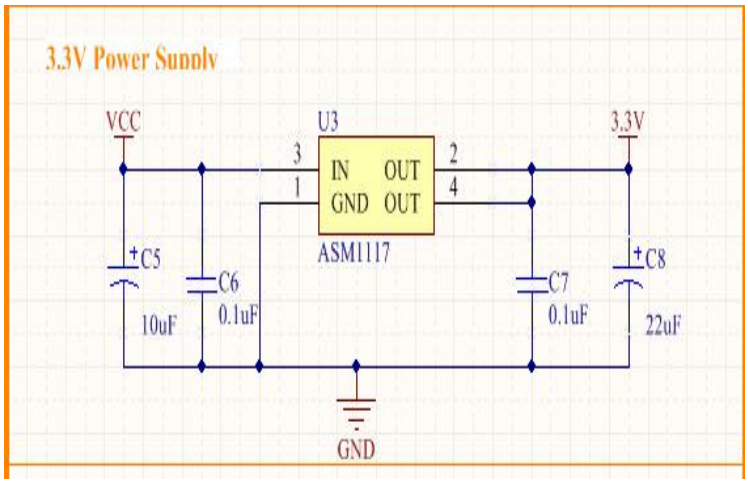



图 3.5 3.3V 供电原理图

### 3.3.3 MCU、蓝牙、摄像头供电模块

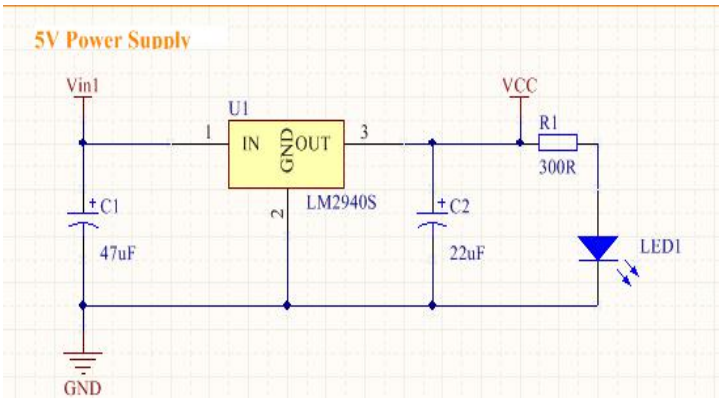



图 3.6　5V 供电原理图

### 3.3.4 驱动芯片供电模块

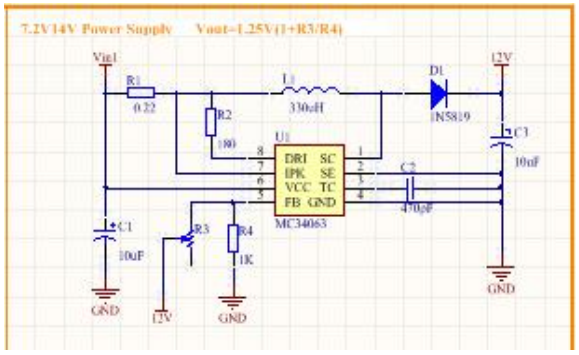



图 3.7　12V 升压原理图

## 3.4 电机驱动模块

电机驱动方面采用的是 mos 管驱动。其优点是使电源端到接地端不会有直接导通的路径，大量节省了电流或功率的消耗，也降低了集成电路的发热量。驱动芯片选择上用的是 IR2104S。使用两片 IR2104s 组成全桥，同时增加一片隔离芯片 741s244，其目的是为了提高驱动能力，同时隔离驱动芯片和单片机，保护驱动芯片和单片机芯片，防止驱动芯片烧坏后将电池电压直接输入到单片机，进而烧坏单片机引脚。驱动原理图如图 3.8 所示：

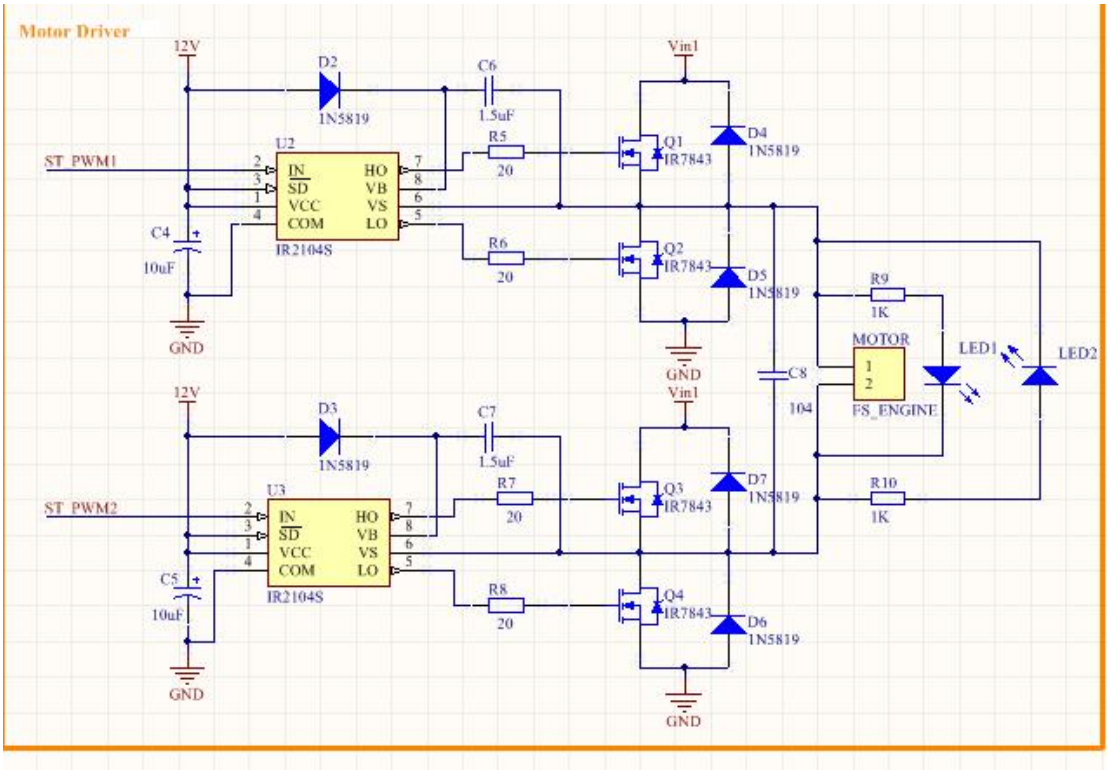



图 3.8 MOS 管驱动电路原理图

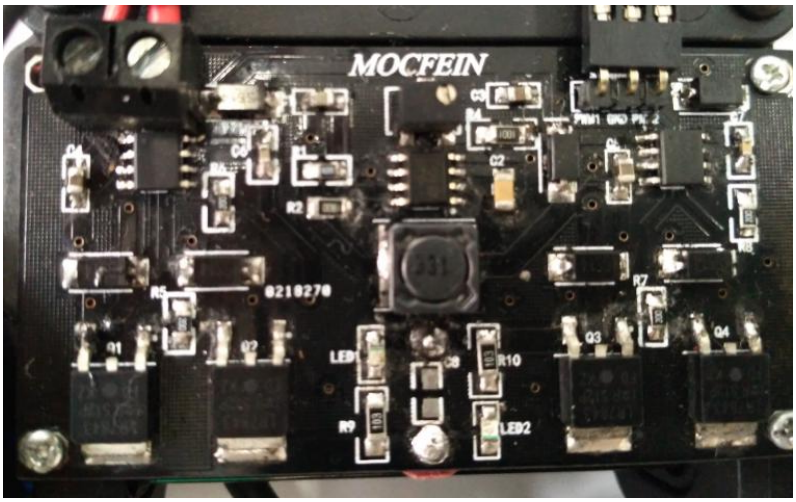



图 3.9 驱动电路实物图

## 3.5 速度检测模块

测速检测方法可以有多种，本设计采用最常用的编码器测速。原理图如下：

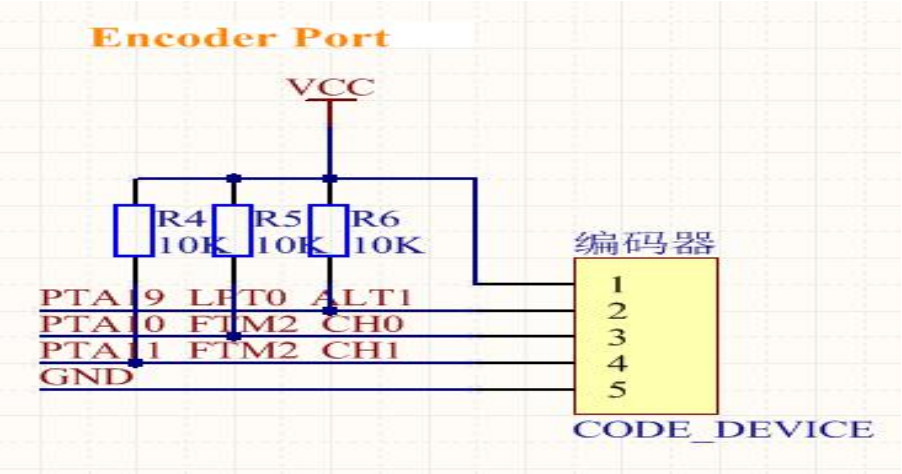



图 3.10 编码器

## 3.6 蓝牙模块

智能车调试过程中，经常需要查看赛道动态信息，特别是摄像头，图像处理过程中需要通过上位机获取赛道信息进行分析处理。小车行进过程中的实时参数，通过串口实现与上位机之间的通信显然是不太现实。于是给小车增加蓝牙通信模块，方便在调试时查看赛道信息和动态参数。

图 3.11 HC-06 无线蓝牙串口透传模块

## 3.7 液晶显示模块

液晶模块能够实时显示各个参数及采集到的赛道图像，通过按键还可以修改各种参数，所以在调试小车的时候使用液晶模块是十分方便有效的。

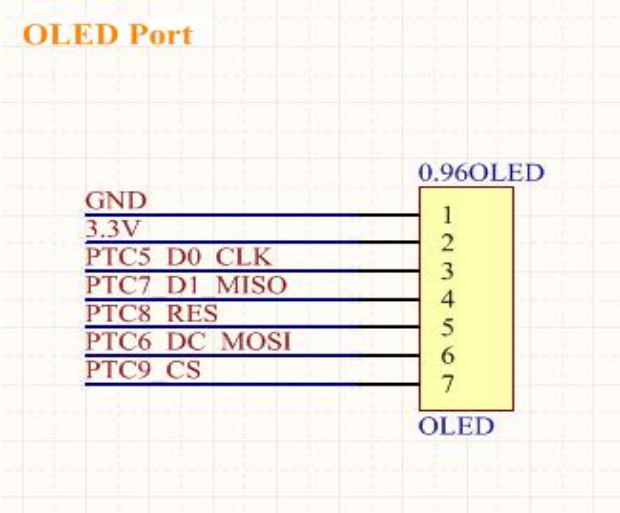



图 3.12　液晶显示模块原理图

## 3.8 其他模块

拨码开关模块用于设定多个额方案，参数以及档位，不仅在调试过程中可以提高调试的效率在比赛的时候能够根据比赛情况合理运用档位，取得理想成绩；SD 卡模块和串口用于数据的传输存储和之后的分析。MMA7455 加速度计模块用于对速度的辅助测量和坡道的检测，使智能车控制更加精确。蜂鸣器模块在调试过程中用于判断检测赛道元素是否精确，作用十分明显。

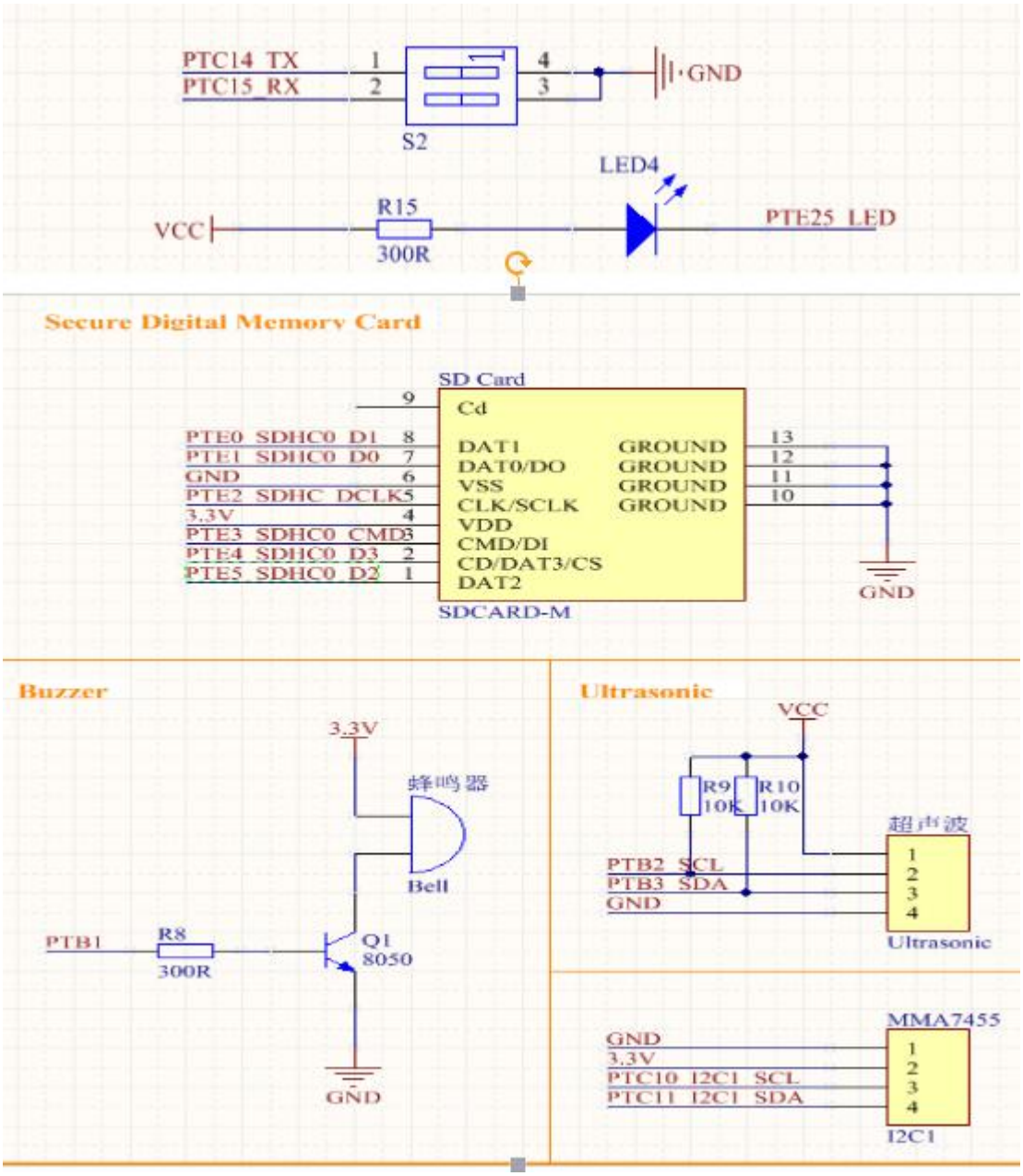

PTC14_TX
PTC15_RX
S2
GND
LED4
R15
300R
VCC
PTE25_LED
Secure Digital Memory Card
SD Card
Cd
PTE0_SDHC0_D1
PTE1_SDHC0_D0
GND
PTE2_SDHC_DCLK
3.3V
PTE3_SDHC0_CMD
PTE4_SDHC0_D3
PTE5_SDHC0_D2
DAT1
DAT0/DO
VSS
CLK/SCLK
VDD
CMD/DI
CD/DAT3/CS
DAT2
GROUND
SDCARD-M
Buzzer
3.3V
蜂鸣器
Bell
PTB1
R8
300R
Q1
8050
Ultrasonic
VCC
R9
10K
R10
10K
超声波
PTB2_SCL
PTB3_SDA
Ultrasonic
MMA7455
PTC10_I2C1_SCL
PTC11_I2C1_SDA
I2C1

图 3.13 其他调试模块

## 3.9 本章小结

硬件电路的可靠运行是整个系统正常工作的基础，本章详细介绍了核心系统板和控制电路板的设计，重点介绍了电机驱动电路、舵机驱动、电源管理电路的原理和设计方法。

# 第四章　软件系统设计及实现

稳定的程序是智能车高速平稳自动寻线的基础。本设计的智能车采用 CMOS 摄像头进行图像的采集和二值化，图像的采集和赛道信息的提取是整个程序的核心部分，只有在提取到正确信息的情况下我们才能对智能车作出精确有效的控制。而在智能车的转向和速度控制方面，参考往届的经验，使用了鲁棒性很好位置式 PID 控制算法，配合使用理论计算和动态参数补偿的办法，使智能车能够稳定快速寻线并作出相应的控制。

## 4.1 图像处理部分

### 4.1.1 赛道边缘提取

在单片机采集图像信号后需要对其进行处理以提取主要的赛道信息，同时，由于交叉道、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。经过摄像头采集回来的图像如下图：

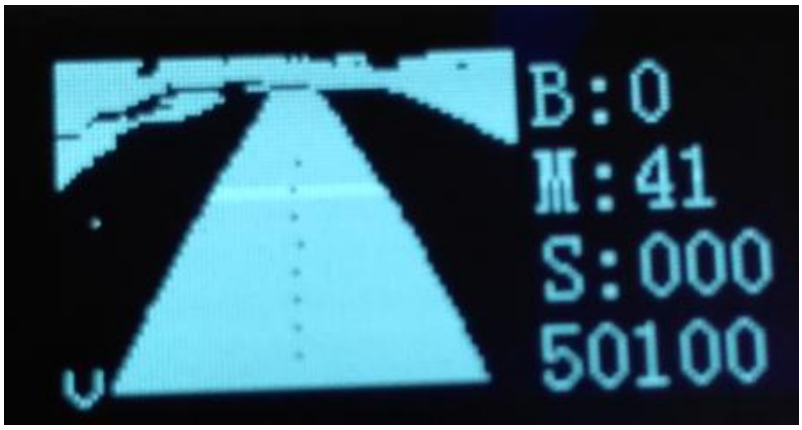



图 4.1 赛道图像

边沿提取的思路和方法列举如下：

(1) 在一场图像内，由近处向远处隔行扫描；

(2) 在一行信息内，中心扩散搜索，根据黑点到白点的跳变提取边缘黑线；

(3) 根据上一行的中点再向两边搜索，根据黑点到白点的跳变提取边缘黑线；

(2) 判断丢线情况并做一些处理，比如十字边界补线，过急弯的情况等；

(3) 同一场图像中，充分利用上一行图像的信息；

(4) 利用近端的信息稳定性较高的特点。

### 4.1.2 赛道信息分析与处理

（1） 当左右边边缘都存在时直接求和平均求出中心值；

（2）只找到一边时，根据图像修正值俩确定中心值；

（3） 两边都没有时，可以认为是在十字交叉处直接将中线置为图像的中线值；

（4）—另外由于图像数据量大，全部扫描一遍会浪费很多时间，因此，只扫描了所用到的 10 行左右的信息，以提高执行的效率。

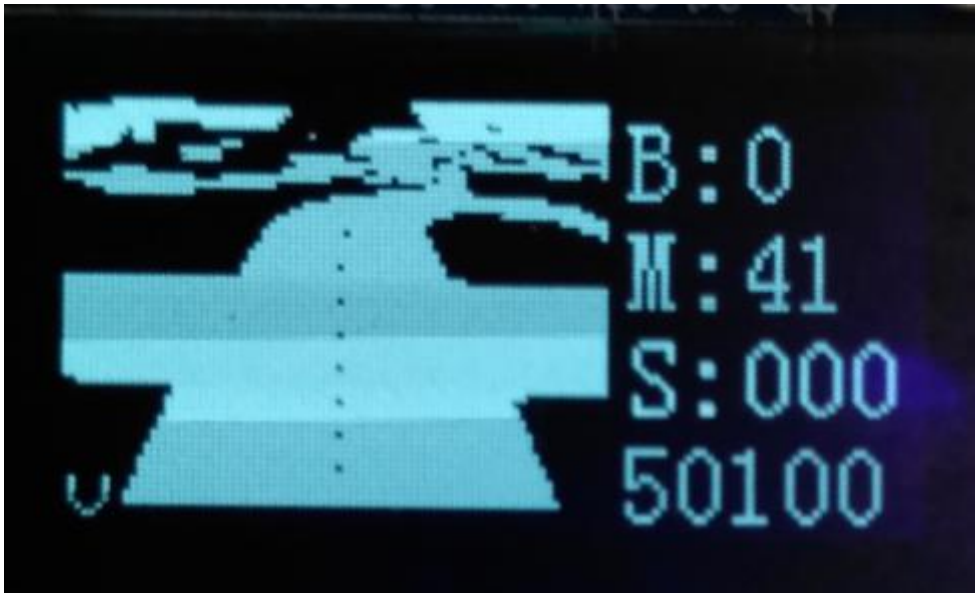



图 4.2（a） 十字弯


图 4.2(b) 小 S 弯

### 4.1.3 障碍、坡道、停车线处理

根据官方规则，带障碍的道路是在直线赛道的基础上，在中心线左侧或右侧添加障碍组成的。路障为黑色的对称三角楔状物，规格为 30x10x5(mm)障碍内侧距离赛道中心线距离为 5 厘米且两侧均有白的。经过多次试验，最终总结出以下几点特征：

（1）障碍前一米内，必定为直道；

（2）障碍存在的地方，赛道宽度会大幅度减少；

（3）障碍存在的地方，赛道宽度存在两次跳变，并且两次跳变的行差值较大。

坡道和起跑线的情况有些类似，在扫描到特征的那几行，赛道宽度都会突变得特别明显，坡道和起跑线的区别是：坡道的赛道宽度突变后不会突变回来，起跑线只是2-3行有明显的赛道宽度变小。据这两个不同的特征，即可解决坡道和障碍物的问题。当检测到坡道时，目标速度降低，使得小车不会因为速度太快而飞起来；当检测到起跑线时，给以0.4s的延迟（因刹车过于灵敏，低速时可能无法冲过停车线），再直接给电机的目标速度为0，小车会通过PID控制，速度骤降为0，这样就不会因为小车提前刹车而浪费时间。


图 4.3（a） 障碍图像


图 4.3(b) 坡道图像

## 4.2 舵机控制

智能车是一个复杂的非线性系统，PID 控制的应用十分广泛。但是有很多不确定因素，导致传统的 PID 算法在智能车转弯方面并不能很好的应用，会出现入弯过慢，出现出弯不正等现象，为了让智能车在转弯时表现出更加优秀的运动性能，通过将 PID 控制中去掉 I 项来加快系统的响应速度，并将二次函数曲

线结合 PD 算法来优化路径，使得小车过弯更加流畅。通过测试发现用 PD 控制来控制舵角可以取得较好的效果。将图象经过算法处理后得到的黑线位置和对应的舵机 PD 参照角度处理成一次线性关系。在 Ki 置零的情况下，舵机在这种动态随动系统对动态响应性能要求更高。更重要的是，通过合理调节 Kp 参数，发现车能在直线高速行驶时仍能保持车身非常稳定，没有震荡，所以基本没有必要使用 Ki 参数。

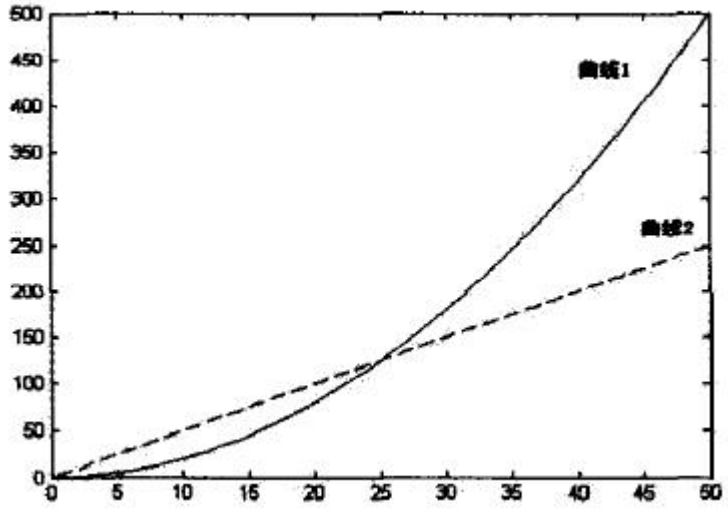



图 4.4 Matalab 模拟比例 P 项曲线

曲线 1 为二次函数型 PD 算法，曲线 2 为普通 PID 算法，可以看出曲线 1 在误差很小的情况下，输出值很小，在误差变大的时候，输出值就变化的很快，很好的解决了弯道跟直道的冲突。

## 4.3 速度控制

要想使智能车实现高速行驶，有四个关键点即对四种赛道类型的控制：1、直线高速行驶。一旦判断是直线后，就要在最短时间内加速到设定速度，并以最高速度行驶。2、入弯提前减速。最关键的处理时在弯道上，弯道最关键的是在入弯。特别是在很长距离直线后的大于 90 度的急转弯，研究发现这是限制最高速度的瓶颈。直线速度越高，与设定的转弯中速度差值就越大，所需减速值就越大，如果等到智能车开始进入弯道才减速已经来不及了，直接后果就是冲出赛道。3 转弯中低速行驶。这里的低速是相对于直线速度而言的。转弯中速度的设定值要根据弯道曲率大小、以及打舵量与速度的配合来调整。4、出弯加

速。一旦判断出是出弯情况后，就要使智能车在最短的时间内加速到接近直线速度的速度值。这与等到完全使出弯道后再提速相比，可以显著提高平均行驶速度。

### 4.3.1 增量式 PID 控速

把占空比大小 PWMFTMCH0 看作油门，PWMFTMCH1 看作刹车。常规的方法是定占空比控速，即直接赋值给 PWMFTMCH0，单片机输出信号不变，即智能车以恒定油门行驶，电机输出的转矩不变，但问题是由于智能车的电机规格不同、整车质量大小不一、加减速时重心位置变化无常、赛道附着力变化复杂，行驶阻力也较难计算，所以很难知道这个油门对应的速度到底是多大，而且当附着力和行驶阻力变化时，特别是直线到弯道的过度阶段，速度波动较大。利用 PID 算法控速，只需设定速度值，可不用管 PWMFTMCH0 到底是多大。

### 4.3.2 ABS 点刹思想

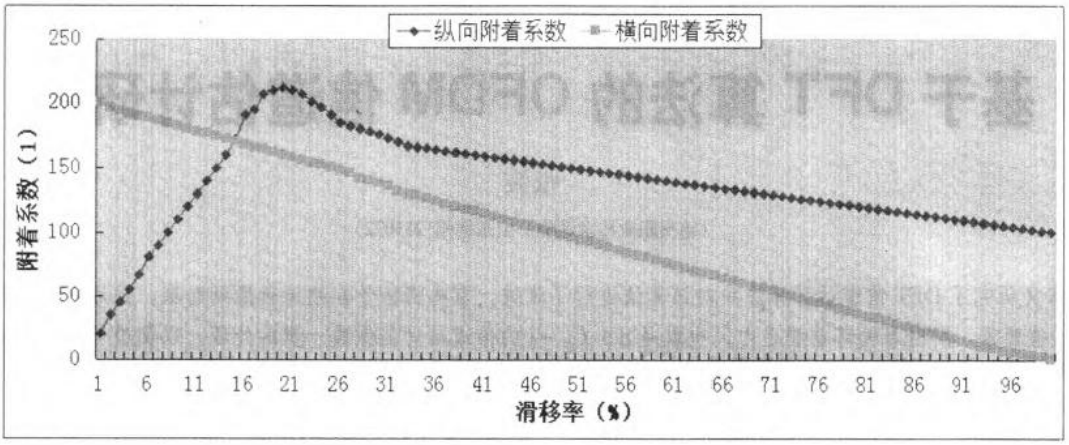



图 4.5 附着系数曲线

实现高速过弯的条件是：速度可控、方向可控。查资料可知汽车横向附着系数、纵向附着系数与滑移率的关系如图 4.5 所示。由图可知在滑移率为 20%左右时，纵向附着系数达到最大，横向附着系数也很大。纵向系数最大，可提供最大制动力，使车速在最短时间内降到设定值。横向附着系数较大，在紧急制动时仍可转动汽车行驶方向。滑移率在 15%-25%是最佳制动范围。在弯道处

理时，常规方法中是否减速只靠赛道类型单一参数，即直线加速、弯道减速。

如果在直线时设定了很高的速度，转弯时才紧急刹车，要想在极短距离内降到转弯中速度，必须输出非常大的反转扭矩，这时的反转扭矩不会使车轮直接反转，只是抵消了车轮的高速转动惯性，使车轮较快减速。此时车轮的滑移率一般都大于 70%，横向附着系数非常小，车子的行驶方向实际上已经不可控了，此时如果转向过小，则直接冲出赛道；转向过大，则直接掉头。所以这种方法只适用于直线速度较低的情况。

当在直线时设定速度很高时，采用了入弯前提前减速以及 ABS 点刹思想。判断是否减速有两个输入参数：赛道类型和车速，即先判断出远端图像是否为入弯类型，如果符合入弯条件，再判断现在车速是否比设定的入弯速度大。如果车速大于入弯速度，再根据速度差值大小选择点刹力度。如果车速小于入弯速度，非但不减速，还要利用 PID 算法使智能车速迅速加速到入弯速度。

刹车时采用循坏点刹的思想。即刹车时先提供一个刹车力度(PWMFTMCH1)，在提供一个油门（PWMFTMCH0），单片机工作频率为 80MHZ。主程序循坏一次时间大概在 0.02ms 左右，刹车和油门在 0.02ms 时间内就可以交替一次。且刹车力度和油门根据速度差值有多种组合。这样就有效避免了车轮抱死的倾向。PID 控制器和点杀控制器为平行关系，程序进入点杀控制器时会自动关掉 PID 控制器，反之则关掉点刹控制器。

## 4.4 PID 算法

经典 PID 算法,适合单输入单输出系统,对于相对复杂的非线性系统可能需要进行补偿所谓的积分饱和现象是指如果系统存在一个方向的偏差，PID 控制器的输出由于积分作用的不断累加而加大，从而导致执行机构达到极限位置，若控制器输出 U(k)继续增大，执行器开度不可能再增大，此时计算机输出控制量超出了正常运行范围而进入饱和区。一旦系统出现反向偏差，u(k)逐渐从饱和区退出。进入饱和区越深则退出饱和区时间越长。在这段时间里，执行机构仍然停留在极限位置而不随偏差反向而立即做出相应的改变，这时系统就像失控一样，造成控制性能恶化，这种现象称为积分饱和现象或积分失控现象。 要使智能车能够平稳快速地运行，需要有高效稳定的控制算法对车模速度进行闭

环反馈控制，于是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即可消除外界各种因素的影响，使得车模运行得更稳定。

PID 控制是工程实际中应用最为广泛的调节器控制规律。问世至今 70 多年来，它以其结构简单、稳定性好、工作可靠、调整方便而成为工业控制的主要技术之一。

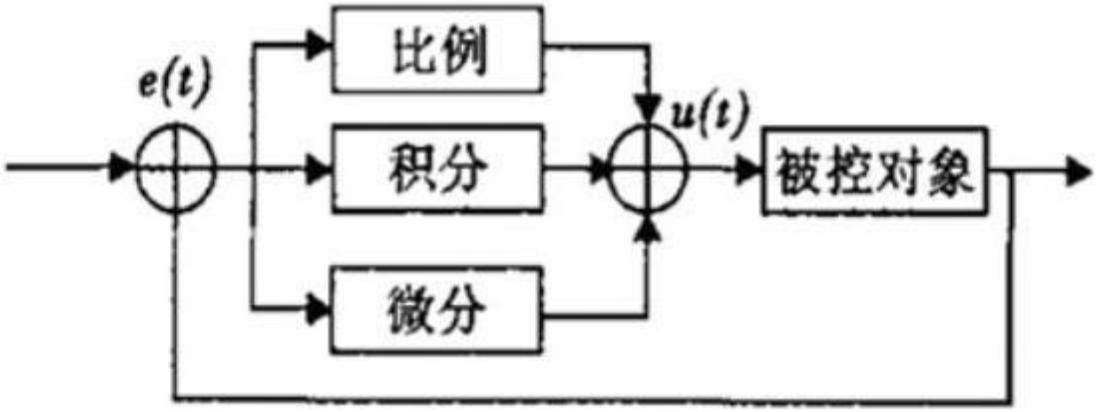



图 4.6 PID 控制原理框图

通常依据控制器输出与执行机构的对应关系，将基本数字 PID 算法分为位置式 PID 和增量式 PID 两种。

(1) 位置式 PID 控制算法 $\Delta e(k) = e(k) - e(k-1)$

基本 PID 控制器的理想算式为

$$u(t) = K_p[e(t) + \frac{1}{T}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}] \qquad \text{公式 1}$$

式中

u(t)——控制器(也称调节器)的输出；

e(t)——控制器的输入(常常是设定值与被控量之差,即 e(t)=r(t)-c(t))；

Kp——控制器的比例放大系数；

Ti ——控制器的积分时间；

Td——控制器的微分时间。

设 u(k)为第 k 次采样时刻控制器的输出值，可得离散的 PID 算式

$$u(k) = K_p e(k) + K_i \sum_{j=0}^{k} e(j) + K_d[e(k) - e(k-1)] \qquad \text{公式 2}$$

式中 Ki 为积分系数，Kd 为微分系数。

(2) 增量式 PID 控制算法 增量式 PID 是指数字控制器的输出只是控制量的增量 Δu(k)。采用增量式算法时，计算机输出的控制量 Δu(k)对应的是本次执行机构位置的增量，而不是对应执行机构的实际位置，因此要求执行机构必须具有对控制量增量的累积功能， 才能完成对被控对象的控制操作。执行机构的累积功能可以采用硬件的方法实现；也可以采用软件来实现，如利用算式程序化来完成。

由式可得增量式 PID 控制算式

$$u(k) = u(k-1) + \Delta u(k)$$

$$\Delta u(k) = u(k) - u(k-1) = K_p \Delta e(k) + K_i e(k) + K_d[\Delta e(k) - \Delta e(k) - \Delta e(k-1)] \qquad \text{公式 3}$$

式中 $\Delta e(k) = e(k) - e(k-1)$。

比较这两种算法不难发现增量式算法算式中不需要累加。控制增量 Δu(k)的确定仅与最近 3 次的采样值有关,容易通过加权处理获得比较好的控制效果。但是增量式是计算机的累积过程，由硬件或被控对象完成，而我们的硬件不具备这样的功能，且增量式 PID 调节是在一个固定值的上下范围内调整，不利于智能车迅速的实施加减速控制，因此速度控制采用位置式 PID 算法。

在连续控制系统中，按偏差的比例（P）、积分（I）、微分（D）进行控制的 PID 控制器获得了广的应用，它的结构简单，参数易于调整，适应性强，对于那些控制模型不准，参数变化较大的被控对象，采用 PID 控制器往往能得到满意的控制效果。

用计算机算法来代替模拟式 PID 控制的数字 PID 控制算法不断改进和完善， 显著地扩展了它的功能。本例采用增量式数字 PID 程序以 PWM 方式来对直流电机进行调速。

增量式数字 PID 调节的数学表达式。其中 Kp 为比例常数，Ti 为积分时间常数，Td 为微分时间常数，T 为采样周期。

$$D(z) = \frac{U(z)}{E(z)} = \frac{a_0 - a_1 z^{-1} + a_2 z^{-2}}{1 - z^{-1}} \frac{\sum SensorRights}{\sum SensorNumber}$$

$$a_0 = k_p(1+\frac{T}{T_i}+\frac{T_d}{T})$$

$$a_1 = K_p(1+2\frac{T_d}{T})$$

$$a_2 = K_p\frac{T_d}{T}$$

公式 4

对位置式算式加以变换，可以得到 PID 调节算法的另一种实用形式（增量算式）

$$\Delta u_n = u_n - u_{n-1} = k_p[(e_n - e_{n-1}) + \frac{1}{T_I}e_n + \frac{T_D}{T}(e_n - 2e_{n-1} + e_{n-2})]$$

公式 6

$A = K_p$ ， $C = K_p\frac{T_D}{T}$ ， $B = K_p\frac{1}{T_I}$ ，

可以得到一个方程，这个方程，经常用来在计算机上做逻辑运算。

增量式 PID 具有以下优点：

(1) 由于计算机输出增量，所以误动作时影响小，必要时可用逻辑判断的方法关掉。

(2) 手动/自动切换时冲击小，便于实现无扰动切换。此外，当计算机发生故障时，由于输出通道或执行装置具有信号的锁存作用，故能保持原值。

(3) 算式中不需要累加。控制增量△u(k)的确定仅与最近 k 次的采样值有关，所以较容易通过加权处理而获得比较好的控制效果。 在使用过程中我们发现传统的增量式 PID,车速控制不及时,容易超调,积分饱和。

最终我们选择的是增量式的 PID，能够很好的满足智能车的加减速以及停车问题。

# 第五章 系统联调

## 5.1 IAR 在线调试工具与 Jlink

程序开放在 IAR Embedded Workbench IDE 下进行，Embedded Workbench for ARM 是 IAR Systems 公司为 ARM 微处理器开发的一个集成开发环境(下面简称 IAR EWARM)。比较其他的 ARM 开发环境，IAR EWARM 具有入门容易、使用方便和代码紧凑等特点。

EWARM 中包含一个全软件的模拟程序(simulator)。用户不需要任何硬件支持就可以模拟各种 ARM 内核、外部设备甚至中断的软件运行环境。从中可以了解和评估 IAR EWARM 的功能和使用方法。

IAR EWARM 中包括集成开发环境 IDE、处理器专家、全芯片仿真、可视化参数显示工具、项目工程管理、C 交叉编译器、汇编器、链接器以及调试器。其中在本设计中重要的部分就是集成开发环境和调试器。

IAR 软件自带在线调试功能，在 J-link 下载完成后可以对智能车的各个参数进行直观准确的观察。


图 5.1 IAR 开发界面和 jlink 仿真器

## 5.2 无线调试蓝牙模块及蓝牙上位机

蓝牙信号的收发采用蓝牙模块实现,具有片内数字无线处理器DRP(Digital Radio Processor)、数控振荡器，片内射频收发开关切换，内置ARM7嵌入式处理器等。接收信号时，收发开关置为收状态，射频信号从天线接收后，经过蓝牙收发器直接传输到基带信号处理器。基带信号处理包括下变频和采样，采用零中频结构。可以提供双工的通用串口，可以方便地和PC机的RS232通信。使用蓝牙与电脑上位机连接，可以很方便的查看摄像头传回来的图像和电机的波形。

此外,针对智能车的特性,采用eclipse软件自主开发了一款手机蓝牙APP：Smart Mocfein，便于查看和更改小车在调试过程中的的档位和参数，明显的节约了整定参数的时间，提高了对智能车的调试效率。

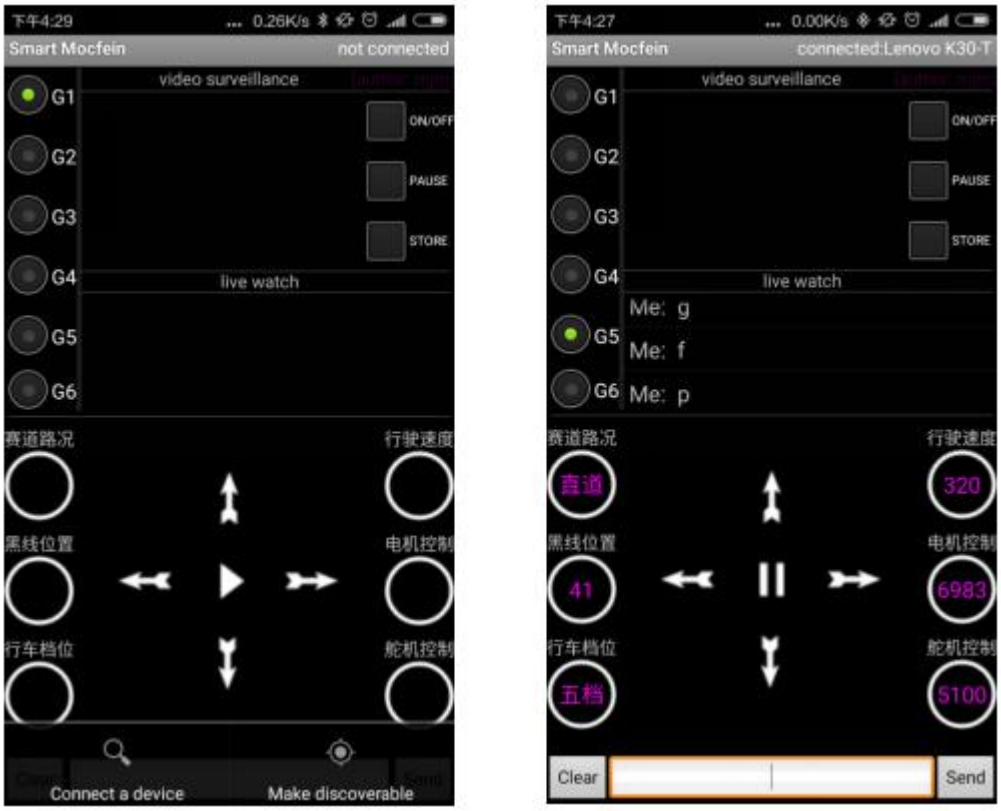



图 5.2 手机蓝牙APP界面

## 5.3 按键及液晶辅助调试

为了调试方便，设计了按键加 OLED。这样一来，可以方便观看和调试参数及图像。可在赛场上快速切换速度模式以适应比赛，还安装了拨码开关。

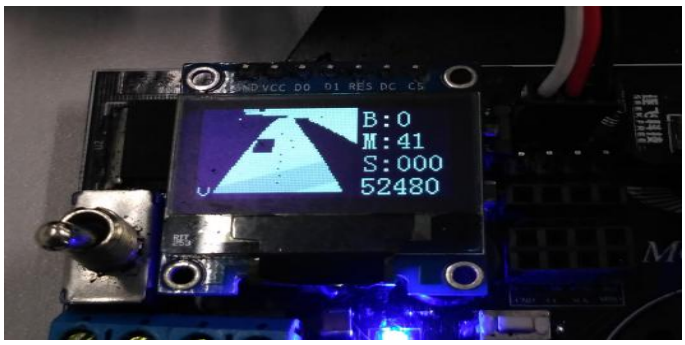



图 5.3 OLED 显示图像及参数

## 5.4 SD 卡调试模块

### 5.4.1 SDHC 简介

SDHC，全名“Secure Digital High Capacity”，是 SD 卡协会（SD Card Association）在 2006 年 3 月发表的 Secure Digital 高容量版本。SD 卡协会有强制规定，所有符合 SDHC 规范的设备都必须标明“SDHC”标志。

### 5.4.2 SDHC 特点

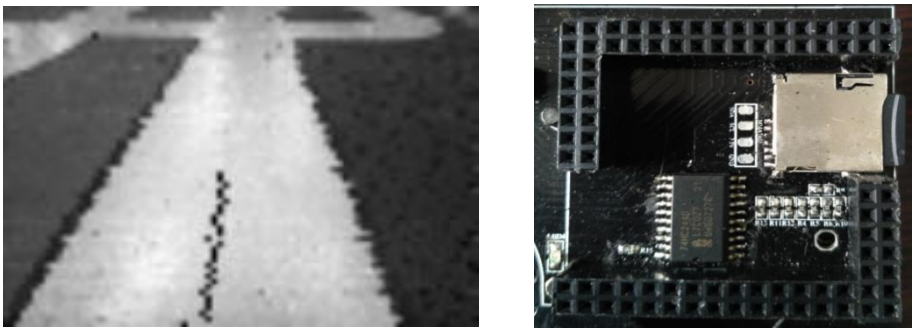
SDHC最大的特点就是高容量（4GB-32GB）。另外，SD协会规定SDHC必须采用FAT32 文件系统，这是因为之前在SD卡中使用的FAT16文件系统所支持的最大容量为2GB，并不能满足SDHC的要求。

作为SD卡的继任者，SDHC主要特征在于文件格式从以前的FAT12、FAT16提升到了FAT32，而且最高支持32GB。同时传输速度被重新定义为Class2、Class4、Class6等级别，高速的SD卡可以支持高分辨视频录制的实时存储。

SDHC卡的外形尺寸与SD卡一样，著作权保护机能等也和以前相同，但是由于文件系统被变更，以前只支持FAT12/16格式的SD设备存在不兼容现象，而支持FAT32(SDHC)的机器，仍可以读取现存的FAT12/16格式的SD卡。

所有大于2G容量的SD卡必须符合SDHC规范，规范中指出SDHC至少需符合Class 2的速度等级，并且在卡片上必须有SDHC标志和速度等级标志。

SD卡一般支持两种操作模式：1、SDHC模式；2、SPI模式。在智能车调试过程中，通过SDHC挂载FATFS文件系统，保存图像数据。



**（a）** **(b)**

图 5.4 （a）SD 卡中保存的图像 （b）SD 卡接口

# 第六章 模型车主要技术参数说明

表 6.1 系统整体参数

| 项目 | 参数 |
| --- | --- |
| 路径检测方法（赛题组） | CMOS 摄像头 |
| 车模几何尺寸（长、宽、高）（毫米） | 295mm/175mm/330mm |
| 前轮轮距、后轮轮距（毫米） | 115mm/95mm |
| 电路电容总量（微法） | 165μF |
| 车模平均电流（匀速行驶）(毫安) | 2000mA |
| 传感器种类及个数 | 摄像头×1　光电编码器×1 |
| 赛道信息检测空间精度（毫米） | 4mm |
| 赛道信息检测频率（次/秒） | 75 |
| 主要集成电路种类/数量 | OV7725 摄像头×1<br>K60 核心板×1 |
| 车模重量（带有电池）（kg） | 1.281 kg |

# 第七章 总结

智能车竞赛是一个多学科、综合性的比赛，其中设计涉及了自动控制、传感技术、模式识别、汽车电子、机械原理、等多个学科。在整个的备赛的过程中，我们不仅仅把所学的理论知识应用于实际，还需要自学大量的新知识。不仅开拓了视野，增加了知识的深度与广度，而且在动手能力、运用知识的能力、分析解决问题方面的能力也有了很大的提升。

技术报告中，我们主要介绍了备赛时的基本思路。从电路方案的设计，机械结构的改造以及图像处理和控制算法方面予以详细介绍。在这个过程中，我们查阅了大量的书籍和资料，并仔细阅读了多份技术报告，结合前人的经验也作出了一些创新，并取得了一些效果。在整个制作过程中，我们始终抱着严谨的态度和求实的心态，遇到问题首先排除低级错误，再检查车模、查阅资料发现错误的根源并予以解决，每一个问题的解决，都充斥着我们的汗水和努力，正是这一个个问题的成功解决，才是我们的小车能够不断突破原来的速度，打破瓶颈。小车的制作过程中往往会遇到很多麻烦，以我们的知识面可能还不能很好的解释，感谢这个过程中老师和学长对我们的帮助和关心，如果没有往届学长的指导和帮助，我们肯定会走不少弯路，有很难有机会去踏足国赛的赛场。参加一次这样的比赛，收获的不仅仅是专业知识和能力，这些更是经历，是以后一份永远抹不去的美好的记忆。

祝愿所有为智能车付出过的小伙伴都能拿到令自己满意的答卷。

# 参考文献

[1]卓晴,黄开胜,邵贝贝等编.学做智能车—挑战“飞思卡尔”杯[M].北京:北京航空航天大学出版社,2007.

[2]谭浩强著．C 程序设计[M]．北京：清华大学出版社，2003.

[3]童诗白，华成英．模拟电子技术基础 [M]．北京: 高等教育出版社，2001.

[4]黄开胜，金华民，蒋狄南，韩国智能模型车技术方案分析[J]，电子产品世界.

[5]邓惜仁, 宋学树. 基于 PID 算法和 ABS 点杀思想的智能车高速行驶研究[J]. 软件, 2013, 34(5):60-63.

[6]何西海,胡延忠. (1997). 汽车主销后倾角内倾角测量原理分析[J]. 汽车实用技术(4), 15-17.

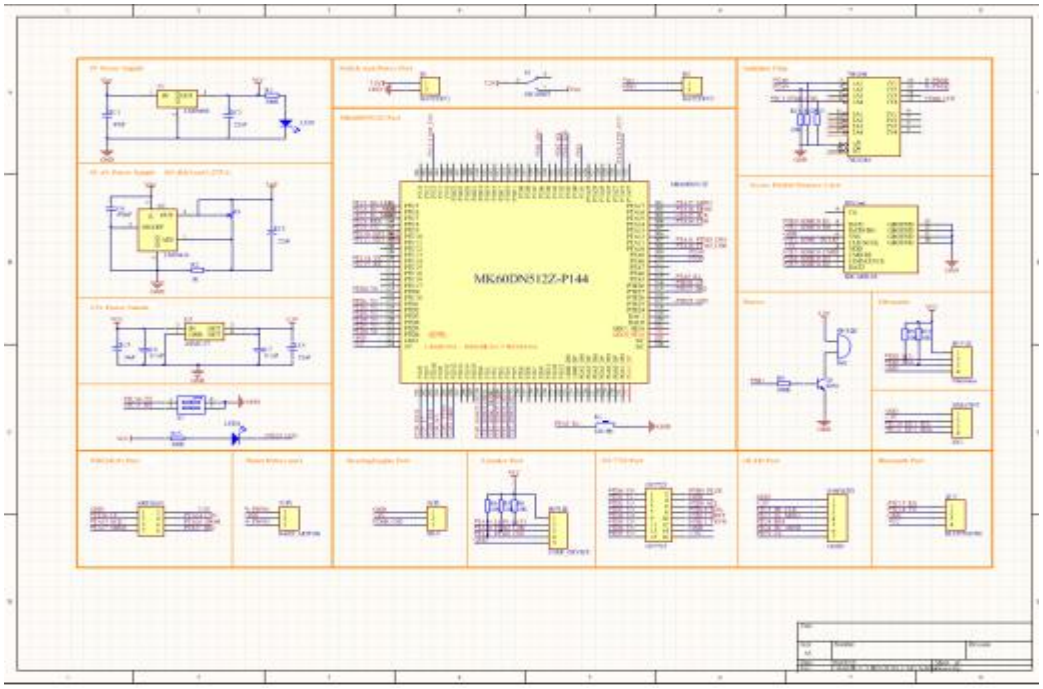
[7]雷贞勇, 谢光骥. 飞思卡尔智能车舵机和测速的控制设计与实现[J]. 电子设计工程, 2010, 18(02):91-92.

[8]高孟杰, 胡晓燕. 基于 K60 竞赛用智能汽车的路径识别系统研究[J]. 科教导刊:电子版, 2016(8):149-150.

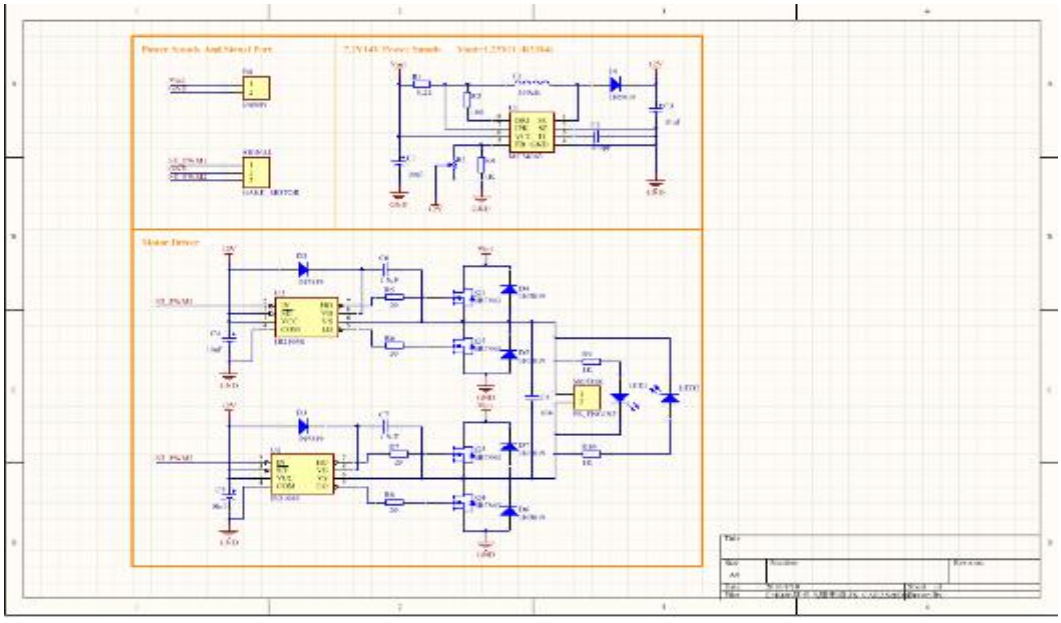
[9]汪波. 智能小车避障与路径优化研究[D]. 重庆理工大学, 2015.

[10]张男, 毛琼. PID 模糊控制器的设计及在智能车转向控制上的应用[J]. 科技风, 2012(5):66-67.

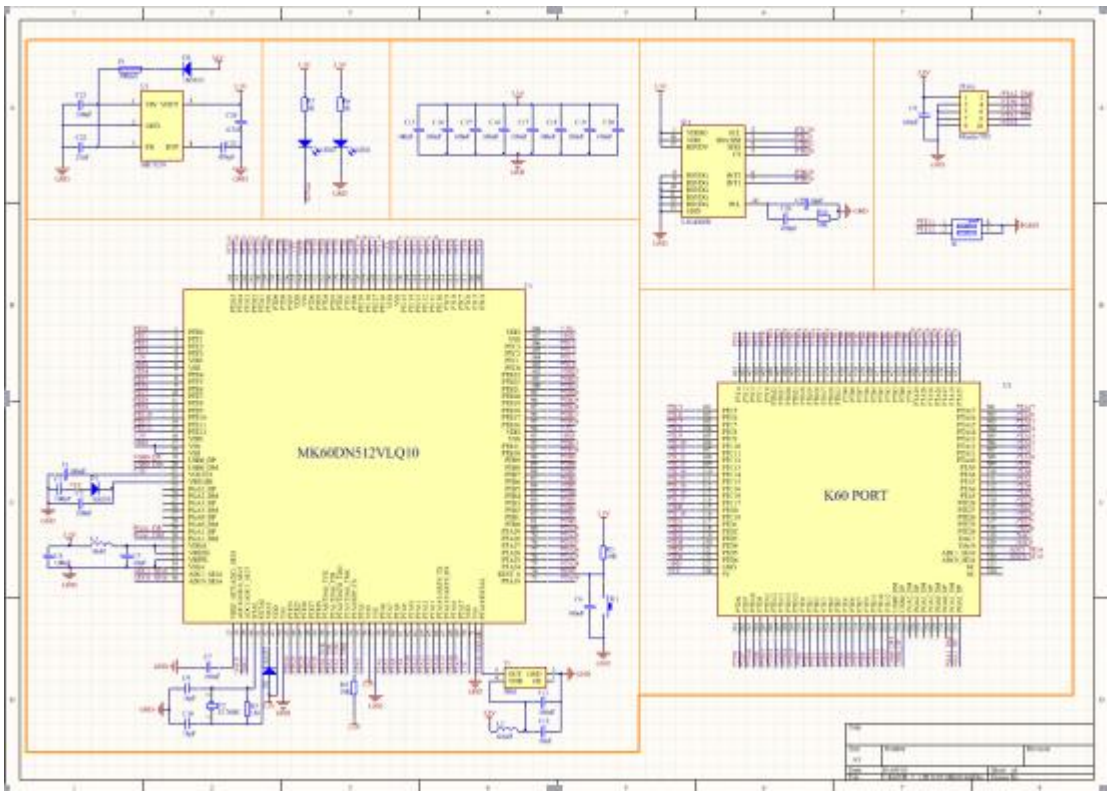
# 附录 A：电路原理图




图（一） 主控板原理图



图（二） 电机驱动原理图




图（三） K60 核心板原理图

# 附录 B：源代码

```
#include "include.h"

struct Blackline Black = {{0},0,0,0,25};
struct MotorPID Motor = {0,0,0};
struct ServoPD Servo = {5010};

void PIT0_IRQHandler(void);                    //定时中断函数声明
void PIT1_IRQHandler(void);                    //定时中断函数声明
//void uart4_handler(void);
//void PORTA_IRQHandler(void);                 //PORTA 端口中断服务函
数

//int MMA_X;
//char ch;

void BellRing(void);
void OLED_Disp(void);
//void Bluetooth(void);
//void SDHC_Model(void);

uint16 Block_Time = 0;
int Block_Dis = 0;


int Stop_Check = 0;
int Stop_time = 0;

extern int right_block;
extern int left_block;
```

```
//=============================================================//
// ****************     ----------主函数----------     ****************//
//                      功能说明：确保智能车安全快速行驶                    //
//=============================================================//
void    main(void)
{
    gears_init();                                       //车行驶档位初始
化

    ov7725_init();                                      //初始换摄像头
    OLED_Init();                                        //OLED 初始化

    //mma7455_init();

    ftm_quad_init(FTM2);                    //初始化  FTM2  为正交解码模式
    ftm_quad_clean(FTM2);

    pit_init_ms(PIT0, 7);
    set_vector_handler(PIT0_VECTORn ,PIT0_IRQHandler);
    enable_irq (PIT0_IRQn);

    pit_init_ms(PIT1, 5);
   set_vector_handler(PIT1_VECTORn ,PIT1_IRQHandler);
    enable_irq (PIT1_IRQn);

    //uart_init(UART4, 9600);
    //set_vector_handler(UART4_RX_TX_VECTORn,uart4_handler);
    //uart_rx_irq_en (UART4);

    //uart_init(UART0, 115200);

    //port_init(PTA5, ALT1 | IRQ_FALLING | PULLUP );
   //set_vector_handler(PORTA_VECTORn ,PORTA_IRQHandler);
    //enable_irq (PORTA_IRQn);                          //使能 PORTA 中断

    steer_init();                                       //初始化  舵机
```

```
    motor_init();                                   //初始化  电机

    led_init(LED1);                                 //初始化 LED1
    led_init(LED2);                                 //初始化 LED2

    //led (LED2,LED_ON);                            //LED2  亮

    gpio_init(PTB1, GPO, 0);                        //初始化蜂鸣器

    //SDHC_Model();                                 //使用 SD 卡

    while(1)
    {
        //uart_sendimg(image_bin, sizeof(image_bin));//发送二值化图像到上位机

        Image_Decompression(image_bin,image_dec[0]);        //解压图像

        blackline_deal();                                   //黑线处理

        //dis_bmp(60,80,image_dec[0],0x00);         //OLED 显示二值化图像

        //OLED_Disp();          //OLED 显示赛道中点/档位、车速、舵机 PWM

        //Bluetooth();                              //蓝牙调试智能车
    }
}

//=======================================================//
//                      函数名称：PIT0_IRQHandler        //
//                      功能说明：PIT0 定时中断          //
//=======================================================//
void PIT0_IRQHandler(void)
{
    steer_PD();
    PIT_Flag_Clear(PIT0);
}
```

```
//=============================================================//
//                          函数名称：PIT0_IRQHandler                          //
//                          功能说明：PIT0 定时中断                            //
//=============================================================//
void PIT1_IRQHandler(void)
{

    Motor.RealSpeed = ftm_quad_get(FTM2);
    ftm_quad_clean(FTM2);

    //MMA_X = (int8)mma7455_read_reg(MMA7455_XOUT8);

    motor_control();
                                                    /
    if(Stop_time < 2000)
        Stop_time ++;
    else
        Stop_Check = 1;

    if(right_block == 1 || left_block == 1)
    {
        if(Block_Time < 100)
            Block_Time ++;
        else
        {
            right_block = 0;
            left_block = 0;
        }

    }

    PIT_Flag_Clear(PIT1);
}
```

```
//=========================================================//
//                    函数名称：uart4_handler                //
//                    功能说明：串口 4 接收字符消息 ch        //
//=========================================================//
void uart4_handler(void)
{
    UARTn_e uratn = UART4;

   if(UART_S1_REG(UARTN[uratn]) & UART_S1_RDRF_MASK)
        {
        //用户需要处理接收数据
        uart_getchar    (UART4, &ch);
        uart_putchar    (UART4 , ch);
    }
}

//=========================================================//
//                    函数名称：Bluetooth                    //
//                    功能说明：实现蓝牙调车                 //
//=========================================================//
void Bluetooth(void)
{
    switch(ch)
    {
        case 'p':
            OLED_Print(100,0,"P");
            BellRing();                                  //蜂鸣器响一声
            Stop_Moving();                               //停车
            OLED_Print(100,0,"G");
            ch = '0';
            break;
        case '1':
            OLED_Print(100,0,"1");
            Motor.SpeedMode = 1;                         //① 档
            gears_init();                                //重新设定档位
```

```
        BellRing();                                    //蜂鸣器响一声
        ch = '0';
        break;
case '2':
        OLED_Print(100,0,"2");
        Motor.SpeedMode = 2;                           // ② 档
        gears_init();                                  //重新设定档位
        BellRing();                                    //蜂鸣器响一声
        ch = '0';
        break;
case '3':
        OLED_Print(100,0,"3");
        Motor.SpeedMode = 3;                           // ③ 档
        gears_init();                                  //重新设定档位
        BellRing();                                    //蜂鸣器响一声
        ch = '0';
        break;
case '4':
        OLED_Print(100,0,"4");
        Motor.SpeedMode = 4;                           // ④ 档
        gears_init();                                  //重新设定档位
        BellRing();                                    //蜂鸣器响一声
        ch = '0';
        break;
case '5':
        OLED_Print(100,0,"5");
        Motor.SpeedMode = 5;                           // ⑤ 档
        gears_init();                                  //重新设定档位
        BellRing();                                    //蜂鸣器响一声
        ch = '0';
        break;
case '6':
        OLED_Print(100,0,"6");
        Motor.SpeedMode = 6;                           // ⑥ 档
        gears_init();                                  //重新设定档位
        BellRing();                                    //蜂鸣器响一声
        ch = '0';
```

```
            break;
        case 'x':

            BellRing();                                    //蜂鸣器响一声
            ch = '0';
            break;
        case 'k':

            BellRing();                                    //蜂鸣器响一声
            ch = '0';
            break;
        case 't':

            BellRing();                                    //蜂鸣器响一声
            ch = '0';
            break;
        case 'j':

            BellRing();                                    //蜂鸣器响一声
            ch = '0';
            break;
        case 'y':

            BellRing();                                    //蜂鸣器响一声
            ch = '0';
            break;
        case 'v':

            BellRing();                                    //蜂鸣器响一声
            ch = '0';
            break;
        default:
            led (LED2,LED_ON);                              //LED2  亮
    }
}
```

```
//=========================================================//
//                      函数名称：PORTA_IRQHandler
//                      功能说明：手动切换智能车行驶档位
//=========================================================//
void PORTA_IRQHandler(void)
{
    uint8    n = 5;                                          //引脚号
    if(PORTA_ISFR & (1 << n))                                //PTA5 触发中断
    {
        PORTA_ISFR    = (1 << n);                            //写 1 清中断标志
位

        Motor.SpeedMode ++;                                  //切换不同档位
        if(Motor.SpeedMode > 6)
            Motor.SpeedMode = 0;

        gears_init();                                        //重新设定档位

        BellRing();                                          //蜂鸣器响一声
    }
}
```

Blackline.c：

```
/==========================================================
//                          函数名称：blackline_deal
//                          功能说明：黑线处理
//==========================================================
void blackline_deal()
{
    Black.centre[6] = get_midpoint(6,H+36);
    bend = 6;

    if(ABS(Black.centre[6] - 40) < Black.Limit && line_sum[6] > 12)
    {
```

```
            Black.centre[5] = get_midpoint(5,H+30);
            bend = 5;
            if(ABS(Black.centre[5] - 40) < Black.Limit && line_sum[5] > 11)
            {
                  Black.centre[4] = get_midpoint(4,H+24);
                  bend = 4;

                  if(ABS(Black.centre[4] - 40) < Black.Limit && line_sum[4] > 10)
                  {
                        Black.centre[3] = get_midpoint(3,H+18);
                        bend = 3;
                       if(ABS(Black.centre[3] - 40) < Black.Limit && line_sum[3] >
9)
                        {
                              Black.centre[2] = get_midpoint(2,H+12);
                              bend = 2;
                              if(ABS(Black.centre[2] - 40) < Black.Limit &&
line_sum[2] > 8)
                              {
                                    Black.centre[1] = get_midpoint(1,H+6);
                                    bend = 1;
                                    if(ABS(Black.centre[1] - 40) < Black.Limit &&
line_sum[1] > 7)
                                    {
                                          Black.centre[0] = get_midpoint(0,H);
                                          if(ABS(Black.centre[0] - 40) < Black.Limit
&& line_sum[0] > 6)
                                                bend = 0;
                                    }
                              }
                        }
                  }
            }
      }
      Black.PreBend = bend;
      switch(Black.PreBend)
      {
```

```
        case 0: OLED_Print(100,0,"0"); break;
        case 1: OLED_Print(100,0,"1"); break;
        case 2: OLED_Print(100,0,"2"); break;
        case 3: OLED_Print(100,0,"3"); break;
        case 4: OLED_Print(100,0,"4"); break;
        case 5: OLED_Print(100,0,"5"); break;
        case 6: OLED_Print(100,0,"6"); break;
        }
}
```

Motor.c：

```
//==========================================================
//                          函数名称：motor_control
//                          功能说明：电机控制
//==========================================================
void motor_control(void)
{
    int SpeedCase;
    switch(Motor.SpeedMode)
    {
      case 1:
        TargetSpeed = 70;
        break;
      case 2:
        TargetSpeed = 80;
        break;
      case 3:
        TargetSpeed = 90;
        break;
      case 4:
        switch(Black.PreBend)
        {
            case 0:
              if(ABS(Black.centre[1] - 40) < 10 && ABS(Black.centre[2] - 40) <
10
                  && ABS(Black.centre[3] - 40) < 10 && ABS(Black.centre[5] -
40) < 10)
```

```
                    TargetSpeed = 160;
                else
                    TargetSpeed = 90;
                break;
            case 1:
                if(ABS(Black.centre[2] - 40) < 10 && ABS(Black.centre[3] - 40) <
10
                    && ABS(Black.centre[4] - 40) < 10 && ABS(Black.centre[5] -
40) < 10)
                    TargetSpeed = 90;
                else
                    TargetSpeed = 70;
                break;
            default:
                TargetSpeed = 70;
        }
        break;
    case 5:
        switch(Black.PreBend)
        {
            case 0:
                if(ABS(Black.centre[1] - 40) < 10 && ABS(Black.centre[2] - 40) <
10
                    && ABS(Black.centre[3] - 40) < 10 && ABS(Black.centre[5] -
40) < 10)
                    TargetSpeed = 160;
                else
                    TargetSpeed = 120;
                break;
            case 1:
                if(ABS(Black.centre[2] - 40) < 10 && ABS(Black.centre[3] - 40) <
10
                    && ABS(Black.centre[4] - 40) < 10 && ABS(Black.centre[5] -
40) < 10)
                    TargetSpeed = 120;
                else
                    TargetSpeed = 110;
```

```
                    break;
                default:
                    TargetSpeed = 110;
            }
            break;
        default:
            switch(Black.PreBend)
            {
                case 0:
                  if(ABS(Black.centre[1] - 40) < 10 && ABS(Black.centre[2] -
40) < 10
                      && ABS(Black.centre[3] - 40) < 10 &&
ABS(Black.centre[5] - 40) < 10)
                  {
                      RunTime ++;
                      TargetSpeed = 360;
                  }
                  else
                      TargetSpeed = 100;
                      BrakeTime = 0;
                      SpeedCase = 0;
                      break;
                case 1:
                    if(RunTime > 30 && SpeedCase < 1 && BrakeTime < 60
&& Motor.RealSpeed > 70)
                      {
                          TargetSpeed = -100;
                          BrakeTime ++;
                      }
                      else
                      {
                          TargetSpeed = 100;
                          SpeedCase = 1;
                      }
                      break;
    }
    Speed_ess = TargetSpeed - Motor.RealSpeed;        //目标速度与实际速度偏
```

```
差

    Motor_PID();                                        //电机 PID 控制
}
```
Steer.c：

```
void steer_PD()
{
    int32 Servo_P,Servo_D;

    switch(Black.PreBend)                                                    //
均值滤波
    {
      case 0:
        now_ess = ((Black.centre[2] + Black.centre[3] + Black.centre[4] +
Black.centre[6]) >> 2) - 40;
        Servo_P = 15 * now_ess;
        Servo_D = 36 * (now_ess - pre_ess);
        Black.BendFlag0 = 0;
        Black.BendFlag1 = 0;
      break;
      case 1:
        now_ess = ((Black.centre[1] + Black.centre[2] + Black.centre[3] +
Black.centre[5]) >> 2) - 40;
        Servo_P = 19 * now_ess;
        Servo_D = 56 * (now_ess - pre_ess);
        if(Black.BendFlag0 < 1 && Black.BendFlag1 <= 1)          //入弯
            Black.Limit = 15;
        else                                                     //出弯
        {
            Black.Limit = 30;
            Black.BendFlag0 = 1;
        }
        Black.BendFlag1 = 1;
        break;
    }
    Servo.PWM = 4945 + Servo_P + Servo_D;
```

```
    pre_ess = now_ess;
    ftm_pwm_duty(SD5_FTM, SD5_CH, Servo.PWM);   //舵机控制 pwm 输出
}
```