第七届“飞思卡尔”杯全国大学生
智能汽车竞赛

技 术 报 告

学　　校：南京理工大学紫金学院

队伍名称：紫金摄像头 1 队

参赛队员：季江

施滢

马火

带队教师：孟迎军　李盛辉

关于技术报告和研究论文使用授权的说明

本人完全了解第七届全国大学生“飞思卡尔”杯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，比赛组委会和飞思卡尔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：_____

_____

_____

带队教师签名：_____

_____

日　　期：_____

摘　要

本文介绍了队员们在准备第七届“飞思卡尔” 杯智能车竞赛过程中设计的基于视觉引导的智能汽车控制系统。智能车的车模采用大赛组委会统一提供的仿真车模，硬件平台采用带MC9S12XS128 处理器的S12环境，软件平台为CodeWarrior IDE 开发环境。文中介绍了智能车控制系统的软硬件结构和开发流程。

整个智能车系统的设计与实现包括了车模的机械结构调整、传感器电路的设计与信号的处理、控制算法和策略优化、系统调试等多个方面。通过对比不同方案的优缺点，在保证提高智能赛车的行驶速度和可靠性，我们最终确定了现有的系统结构和各项控制参数。

关键字：智能车，视觉引导，图像处理，路径识别

目　次

第一章　引言

1.1　智能车发展状况

智能车的发展是从自动导引车（Automatic Guided Vehicle,AGV）起步的。AGV是指装有电磁或光学等自动导引装置，能够沿规定的导引路径行驶，具有安全保护及各种移栽功能的运输车辆。1913年，美国福特汽车公司首次将有轨导引的AGV代替输送机用到底盘装配上。1954年，英国采用在地板下埋线，组成了电磁感应导向的简易AGVS。AGVS（Automatic Guided Vehicle System）是指自动导引车（AGV）和地面导引系统组成的、进行物料搬运等作业的光机电一体化的系统。1955年，英国研制出了在生产线上实用的AGVS。1959年，在美国首先出现了应用AGV的自动化仓库。1982年，德国出现了第一辆无人叉车。1985年，计算机通讯。识别技术应用于AGVS。在国内，北京起重机械研究所于1976年研制出了我国第一台AGV。1991年起，中科院沈阳自动化研究所研制了客车装配AGV系统。

随着科学技术的发展，许多新技术都应用到AGV或AGVS上。例如，激光技术的应用使AGV实现虚拟路径的导航和安全保护；无线局域网的应用使AGV的调度实时性更强，是AGV调度技术的一场革命；现场总线的应用使AGV的可靠性和可维护性得到提高，RFID的应用使AGV与地面系统的信息交互量更大。自适应性更强。

同时，随着智能交通系统研究的深入，无人驾驶智能车辆的研究也越来越受到人们的关注。

1.2　智能汽车竞赛介绍

全国大学生“飞思卡尔”杯智能汽车竞赛是在规定的模型汽车平台上，使用飞思卡尔半导体公司的 8 位、16 位微控制器作为核心控制模块，通过增加道路传感器、电机驱动电路以及编写相应软件，制作一个能够自主识别道路的模型汽车，按照规定路线行进，以完成时间最短者为优胜。智能汽车的行驶控制

一直以来是自动化、汽车等学科研究的目标，首届“飞思卡尔”杯全国大学生智能车大赛以邀请赛的方式使更多的学校和同学有了探索的机会。现在历经七届，每一届都较前一届无论是速度还是稳定性都有新的突破。大学生智能模型车竞赛是在飞思卡尔半导体公司资助下举办的以 S12 单片机为核心的大学生课外科技竞赛。组委会将提供一个标准的汽车模型、直流电机和可充电式电池，参赛队伍要制作一个能够自主识别路线的智能车，在专门设计的跑道上自动识别道路行驶，谁最快跑完全程而没有冲出跑道，谁就是获胜者。

为了追求小车的高速和稳定的目的，人们对人工智能与机器人技术，汽车技术，自动控制技术各方面都进行了更广泛、更深入的层面展开研究 ,这样无疑对学术研究和生产应用都有很强的实际意义。比赛涉及到的专业知识有自动控制、模式识别、传感技术、汽车电子、电气、计算机、机械等多个学科，对学生的知识融合和实践动手能力的培养有重大的意义，对高等学校控制及汽车电子学科学术水平的提高，具有良好的长期推动作用。

第二章　系统总体设计

智能车系统的制作要求是能够自主识别路线，即在按规则专门设计的跑道上自动识别道路行驶，要求最快跑完全程而没有冲出跑道，要求智能小车运行又快又稳。因此对于小车的控制系统来说稳定性和快速性是控制系统设计的两个重要指标。

2.1　工作原理

根据需求分析，经过仔细研究，决定采用模块化设计。智能汽车的硬件系统由核心控制模块(MCU)、电源管理模块、传感器模块、存储器模块、电机驱动模块、舵机驱动模块、人机接口模块、无线通讯模块组成。

智能车系统的工作原理示意图如图 2.1 所示：

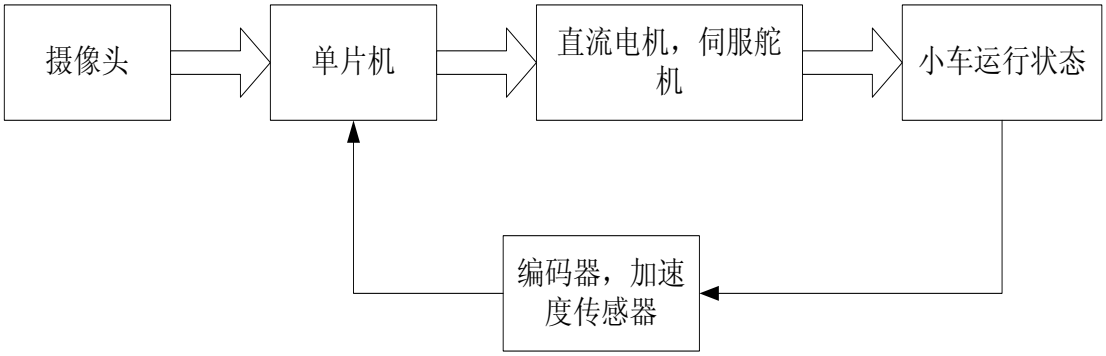

图 2.1　智能汽车系统工作原理示意图

智能汽车系统通过视觉传感器来检测前方的赛道信息，并将赛道信息发送给单片机。同时，通过编码器和加速度传感器构成的反馈渠道将车体的行驶速度及加速度信息传送给主控单片机。根据所取得的赛道信息和车体当前的速度及加速度信息，主控单片机做出决策，并通过 PWM 信号控制直流电机和舵机进行相应动作，从而实现车体的转向控制和速度控制。

2.2　硬件结构

智能车控制系统从硬件上分为电源模块、传感器模块、信号处理模块、直流电机驱动模块、转向伺服电机驱动模块和单片机模块。

系统总体结构如图 2.2 所示：

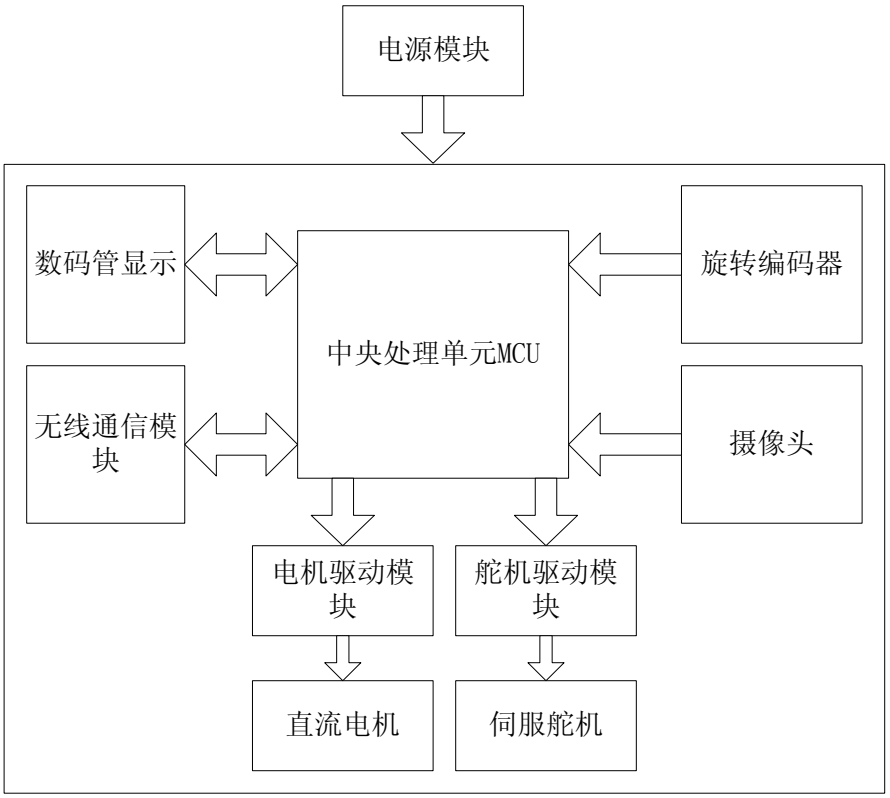

图 2.2　智能汽车控制系统总体设计框图

2.3　软件结构

系统硬件位于底层，是整个系统的基础，系统软件结构则根据硬件和控制需求来制定。

系统基本的软件结构如图2.3所示。由摄像头产生场中断信号，然后单片机开始采集图像、处理图像，同时测出电机速度，最后由所得到的数据对舵机和电机进行控制并将相关数据通过无线模块发送出来。控制周期为摄像头场中断的周期，即1/60s。

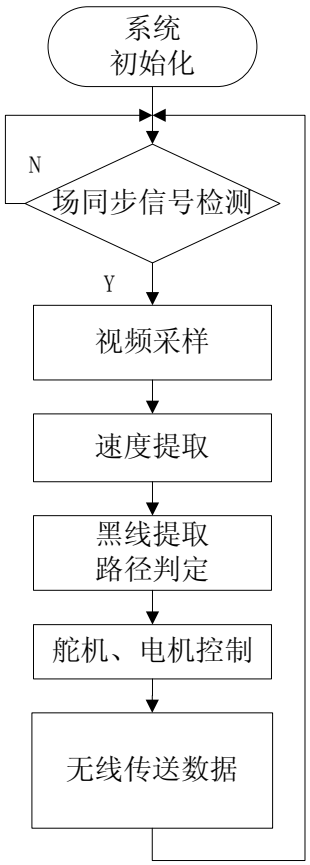

图2.3　系统软件结构

2.4　小结

本章介绍了智能汽车控制系统的工作原理。根据系统的需求简要给出了系统的总体设计方案，并划分了系统的几个模块，。

第三章　机械结构设计与实现

为了使车能够更稳定的高速运行，我们对这次比赛车模进行了系统的分析。今年的车模精度不是很高，因此要尽量在规则允许范围内改造车模，提高车模整体精度是很必要的。另外，我们在实际调试中发现，前轮的束角和主销倾角对车的高速运行下的稳定性影响很大。高速运行下舵机的转动速度对车转向的灵活程度也起到了根本性的作用。此外，由于摄像头是外加的传感器，良好的固定方案能最大限度发挥它的前瞻远，视野广的特点。所以，在整车的机械结构方面我们进行了三方面改进：转向机构改进、摄像头设计安装和前轮束角调整。

3.1　赛车基本参数

此次比赛所用赛车车模是由深圳博思公司提供的A型车模，具体车模数据如表3.1所示：

表3.1　车模参数

基本参数	尺寸
轴距	200mm
轮距	135mm
车轮宽度	24mm
车轮直径	50mm
车长	300mm
车宽	160mm
摄像头高度	300mm

赛车机械结构只使用竞赛提供车模的底盘部分及转向和驱动部分。控制采用前轮转向，四轮驱动方案。车模改装是我们的第一步工作，在严格遵守比赛

规则对车模要求的前提下，车模进行重装和改装。如图3.1为车模外观图。

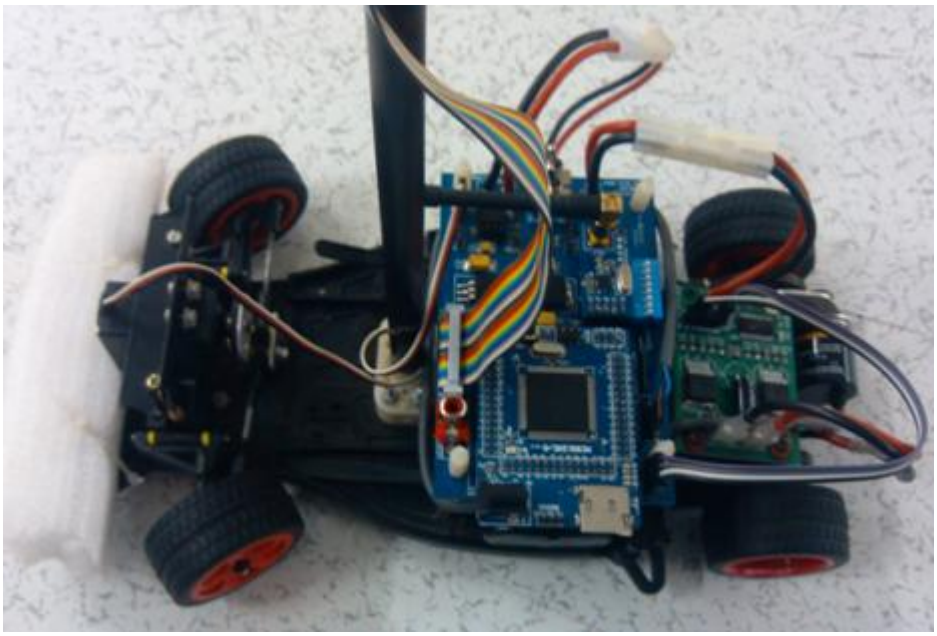

图 3.1　车模整体外观图

3.2　机械结构调整

3.2.1　摄像头的安装

摄像头的安装位置应合适选取。安装位置太低，会导致视域不够广阔，影响寻线的有效范围，特别是在今年比赛规则将引导线贴在赛道两边的情况下，使寻线更加困难。若是将摄像头角度抬高，图像畸变严重；安装位置太高，会将整车重心提高，导致智能车在快速转弯时翻车。安装位置合适的一个标准是：在此位置的拍摄范围能满足控制的需要。由于本届比赛引导线贴在赛道两边，所以摄像头尽量往上提。

3.2.2　前轮倾角调整

前轮定位的作用是保障汽车直线行驶的稳定性，转向轻便和减少轮胎的磨损。前轮是转向轮，它的安装位置由主销内倾、主销后倾、前轮外倾和前轮前束等4 个项目决定，反映了转向轮、主销和前轴等三者在车架上的位置关系。

主销内倾是指主销装在前轴略向内倾斜的角度，它的作用是使前轮自动回

正。角度越大前轮自动回正的作用就越强烈，但转向时也越费力，轮胎磨损增大；反之，角度越小前轮自动回正的作用就越弱。

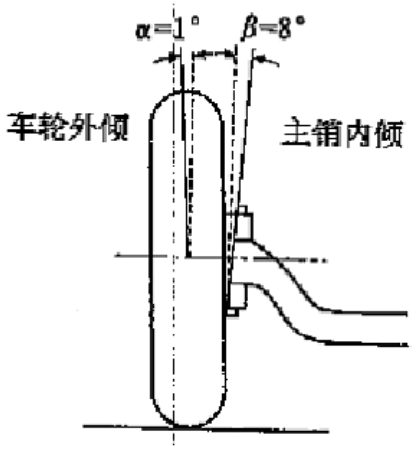

图3.2　前轮外倾角示意图

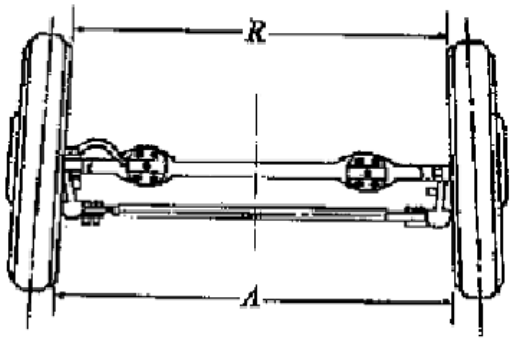

图3.3　前轮约束示意图

主销后倾是指主销装在前轴，上端略向后倾斜的角度。它使车辆转弯时产生的离心力所形成的力矩方向与车轮偏转方向相反，迫使车轮偏转后自动恢复到原来的中间位置上。由此，主销后倾角越大，车速越高，前轮稳定性也愈好。

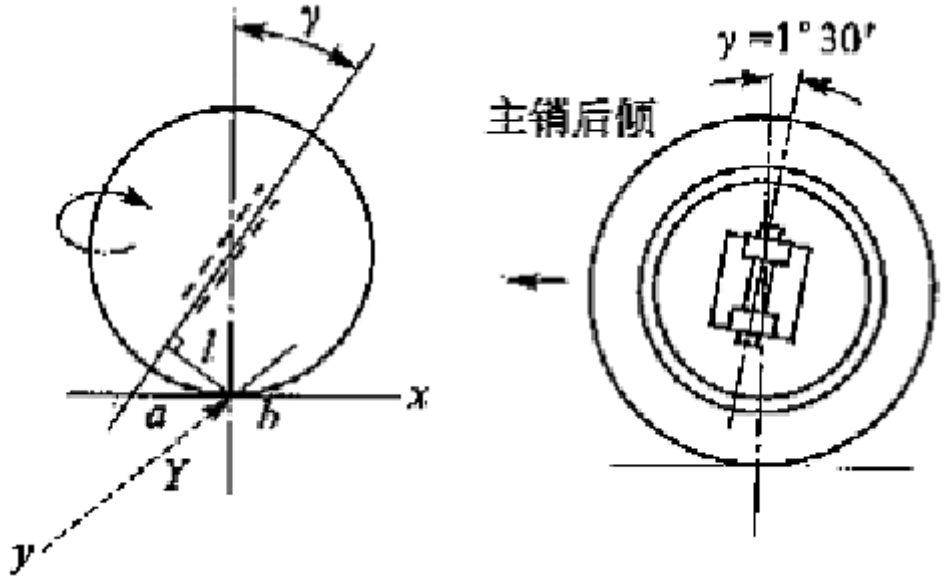

图3.4　主销后倾纠正车轮偏转原理图

前轮外倾角对汽车的转弯性能有直接影响，它的作用是提高前轮的转向安全性和转向操纵的轻便性。

3.2.3　PCB板安装

电路板是这个系统的核心，如何正确的设计和安装是智能车取得良好成绩的关键。我们分别制作了电机驱动板和人机接口板，这样一来，电路板的性能比较稳定而且重量比较轻，保证了智能车能快速稳定的跑完全程。

同时，主控板与各个执行单元的接口也是应该考虑的因素，所以要尽可能照顾到相应模块的接口方向，使接插线的连接不出现交叉或缠绕的现象。

根据小车实际的运行过程可以看出，各执行部件与主板的接口都很恰当，接插线没有缠绕和交叉，该方案达到了良好的效果。

3.3 小结

本章对模型车车体的优化调整进行了介绍。根据汽车理论的基本知识，通过原理分析，对车体姿态参数进行了多方面的调整与优化，并取得了良好的效果。

第四章　系统硬件电路设计

本系统硬件电路的设计目标为：可靠、高效、简洁。可靠性是系统设计的第一要求，我们对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。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，降低车体重心位置，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。

4.1　核心控制器

单片机最小系统板使用 MC9S12XS128 单片机。该芯片采用 5V 供电，功能强大，总线频率高达 40MHz。芯片含有丰富的片内设备，包括 128K 的 Flash 存储器，8K 的 RAM，8K 的 EEPROM，两路串行通信接口（SCI），一路串行外围接口（SPI），八路定时器通道，两个（80 引脚为一个）八路可调转换精度的 A/D 口，八路 PWM 输出，91（80 引脚为 59）个离散数字 I/O 口，一个 MSCAN 模块。该单片机适合于在汽车电子中的应用，成为很多车载电子设备的控制芯片。MC9S12XS128 单片机有 112 引脚和 80 引脚两种型号，前者比后者多出 8 位 AD 口和其他的一些引脚资源。在对小车各个模块进行逐一分析后，确定了单片机的引脚分配。

引脚分配如表 4.1：

表4.1：最小系统引脚分配

摄像头	PP0,PH0,PE[2,3],PA
串口通信模块	PS[0,1]
nRF24L01（无线射频模块）	PA[6,7],PS[4,5,6]
拨码开关	PB[0,1,2,3,4,5,6,7]
编码器	PT7
电机驱动	PWM[5,7]
舵机	PWM3
数码管	PK[0,1,2,3,4,5,6,7]

4.2 电源模块设计

本系统全部硬件电路的电源由 7.2V、2A/h 的可充电镍镉电池提供。由于电路中的不同电路模块所需要的工作电压和电流容量不相同，因此电源模块应该包含多个稳压电路，将充电电池电压转换成各个模块所需要的电压。各模块供电如图 4.1：

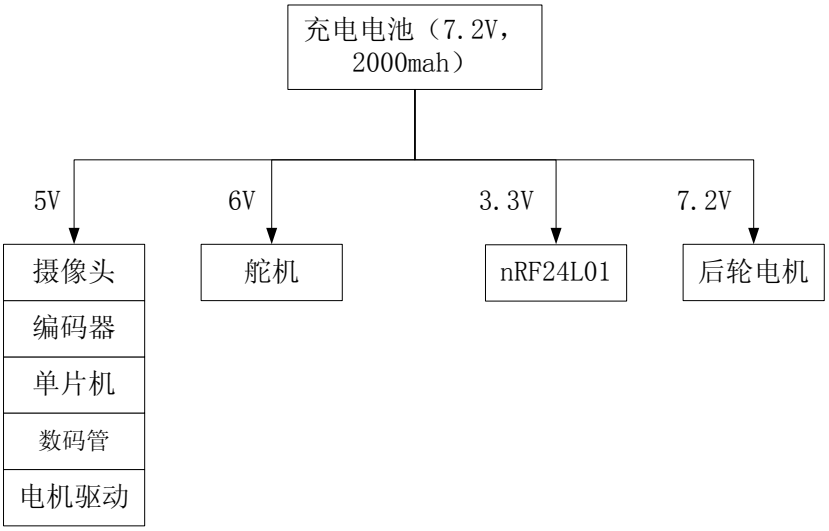

图 4.1　系统供电分析图

这里考虑到人机接口板所需电流为 1A，而单片机等芯片要求电流不高，所以采用两块 5V 稳压芯片分别供电。由图可知，从电压上看，系统的供电电压分为+7.2V、+6V、+5V 和+3.3V 几个档。从功率上看，系统的电源又可以分为信号电源和功率电源两部分。功率电源为电机驱动模块和舵机供电，信号电源则为余下的部分供电。

4.2.1　5V 电源

由上述分析，5V 电源选取两种电源稳压芯片，一种要求纹波小，电流在 1A 以下，工作稳定，给单片机等芯片供电；另一种要求电流在 1A 座右，输出电流纹波没有太高要求，专门为人机接口板的数码管显示驱动和键盘扫描控制芯片供电。

由于整个系统中+5V 电路功耗较小，为了降低电源纹波，我们首先使用串联型稳压电路，另外，后轮驱动电机工作时，电池电压压降较大，为提高系统工

作稳定性，必须使用低压降电源稳压芯片。

为了提高电源的利用率，我们进一步选择 DC/DC 电源稳压电路。DC/DC 是开关型稳压电路，它的优点是电路结构简单，对电源的高频干扰有较强的抑制作用、效率高，输入电压的范围宽，输出电压，电流的纹波值较小。

此外，本系统选用 LM2940 为人机接口板的数码管显示驱动和键盘扫描控制芯片供电。LM2940 也是常用的高性能线性稳压芯片之一，它的工作压差较小，只要输入电压达到 6V 以上就可以稳定输出 5V 电压，同时它的输出电流为 1.5A，满足人机接口板的 1A 要求。连接如图 4.2 所示：

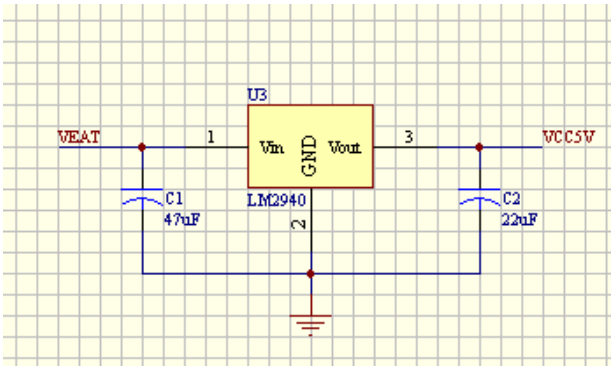

图 4.2　LM2940 电路原理图

4.2.2　6V 电源

由舵机的技术规格可知，其工作电压为 4-6V。由于提高舵机的工作电压可以缩短舵机的反应时间，所以选择其工作电压的上限 6V 来供电。与 5V 电源苛刻的性能要求相比，6V 电源的性能指标要宽松一些，故其选型也比较容易，这里采用 TI 公司的 TPS7350 来实现。

TPS7350 作为系统电源管理芯片有点如下：第一，只需要很少的外围器件即可达到应用系统的要求，实现稳定的电压输出；第二，由于“热损失”小，在设计中基本可以不用考虑电源芯片的散热问题，为电路的设计带来了方便。

为了使舵机的工作性能最佳，在不改变舵机机械结构的基础上，将舵机的工作电压设定为 6V 是一种行之有效的解决办法。然而 TPS7350 的输出电压为 5V，为了实现 6V 输出，需要如图 4.3 在 TPS7350 上加两个二极管，提高对地参考电位，即可满足稳定 6V 输出的要求。经实际使用证明了该方案的可行性。

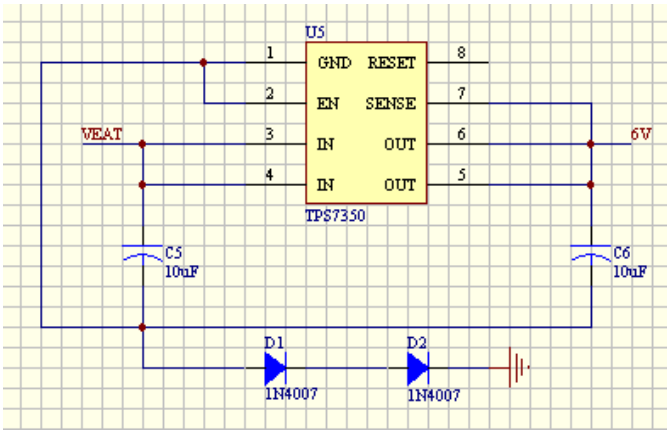

图 4.3　LM1117-ADJ 电路连接图

4.2.3　3.3V 电源

系统中，无线通信模块和加速度传感器需要 3.3V 的供电。由于其电压值较低，故很容易满足而不必考虑电压被拉低的情况。这里采用同样来自于国家半导体公司的 LM1117-3.3，电路如图 4.4 所示。

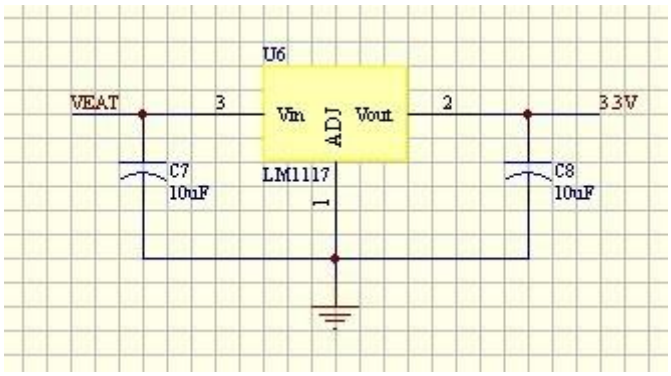

图 4.4　LM1117-3.3 电路连接图

4.3　摄像头模块

4.3.1　摄像头工作原理

OV7620 CAMERACHIPTM 图像传感器，体积小、工作电压低，提供单片 VGA 摄像头和影像处理器的所有功能,如图 4.5。通过 SCCB 总线控制,可以输出整帧、子采样、取窗口等方式的各种分辨率 8 位影响数据。如图 4.5 为 SCCB 时序。该产品 VGA 图像最高达到 30 帧/秒。用户可以完全控制图像质量、数据格式和传输方式。所有图像处理功能过程包括伽玛曲线、白平衡、饱和度、色度等都可

以通过 SCCB 接口编程。OmmiVision 图像传感器应用独有的传感器技术，通过减少或消除光学或电子缺陷如固定图案噪声、托尾、浮散等，提高图像质量，得到清晰的稳定的彩色图像。如图 4.6。

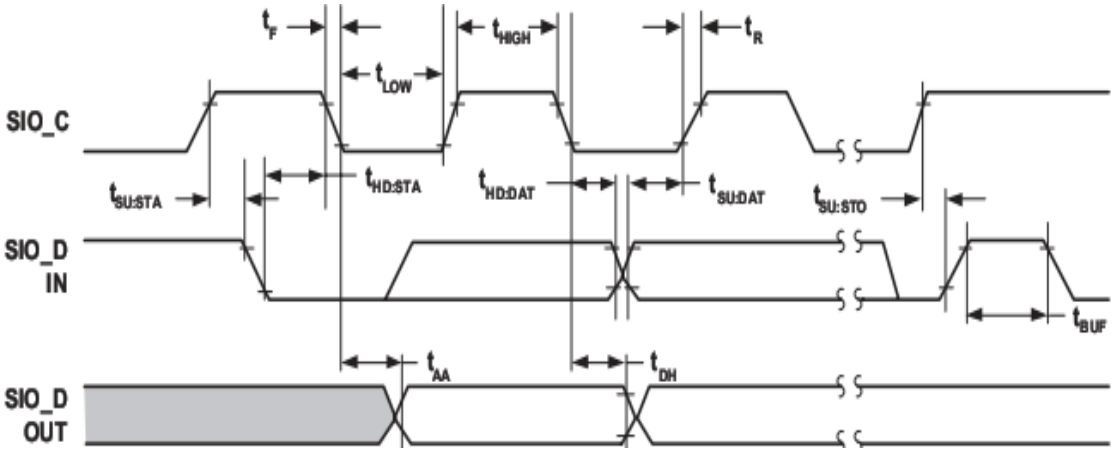

图 4.5　SCCB 时序

图 4.6　摄像头 OV7670

如图 4.7 为 OV7620 行场信号时序图，VSYNC 为图像输出的场中断信号，HSYNC 为行中断信号，HREF 为行输出使能，当 HREF 为高电平时输出一行。当图像分辨率为 640×480 时，两个 VSYNC 之间有 480 个 HSYNC 或 HREF，即代表 480 行，而每两个 HSYNC 之间 HREF 为高电平，并且期间有 640 个 PCLK，代表每行 640 个点。当图像分辨率为 320×240 时同理。

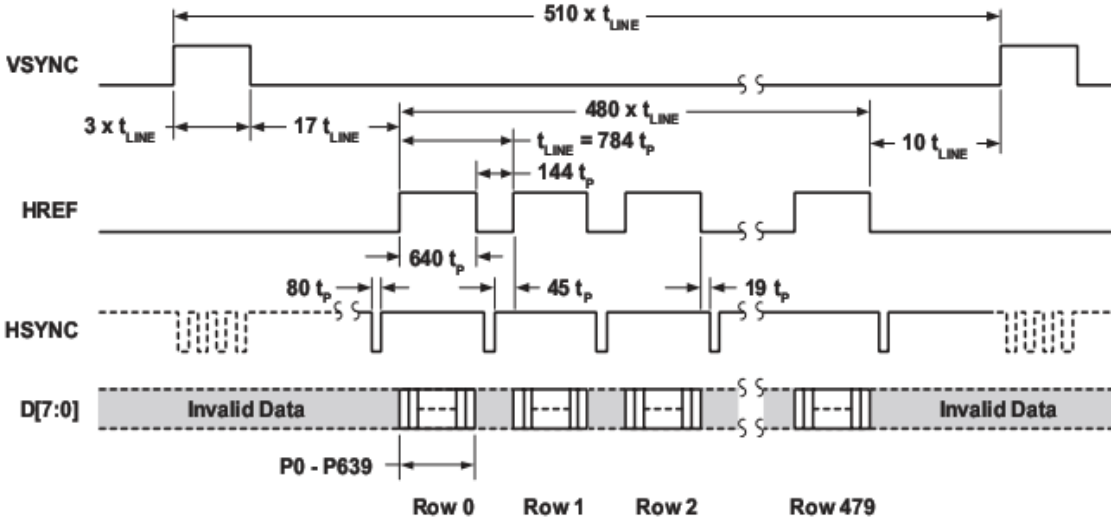

图 4.7　OV7670 行场中断信号时序

4.4　舵机驱动模块

舵机负责智能汽车的转向，舵机模块能否稳定的工作直接影响到智能汽车在赛道上高速行驶时的稳定性以及转向时的灵敏度和精确度。本系统的舵机由大赛组委会统一提供，型号为 S3010。

影响舵机控制特性的一个主要参数是舵机的响应速度即舵机输出轴转动角速度，S3010 型舵机响应速度为 0.1-0.13s/50 度。

控制舵机的脉冲可以使用 MC9S12XS128 的 1 路 PWM 产生。单片机中有 8 路独立的 PWM 输出端口，可以将其中相邻的 2 路 PWM 输出级联成一个 16 位 PWM 输出。在单片机总线频率为 80MHz 的时候，改变 PWM 占空比常数可以改变输出脉冲的宽度。而脉冲信号的宽度决定舵机输出舵盘的角度。

舵机内部电路通过反馈控制调节舵盘角位。由于自身即为角度闭环控制，而且性能较好，故系统中就不必考虑外加舵机闭环。所用舵机如图 4.8 所示。

图 4.8　舵机 S3010

4.5　编码器

为了使得赛车能够平稳地沿着赛道运行，需要控制车速，使赛车在急转弯时速度不至过快而冲出赛道。通过控制驱动电机上的平均电压可以控制车速，但是如果开环控制电机转速，会受很多因素影响，例如电池电压、电机传动摩擦力、道路摩擦力和前轮转向角度等。这些因素会造成赛车运行不稳定。通过速度检测，对车模速度进行闭环反馈控制，即可消除上述各种因素的影响，使得车模运行得更稳定。此外，在记忆算法中为了记录道路信息，需要得到赛车运行距离，这也可以通过车速检测来实现。车速检测的方式有很多种，例如用测速发电机、转角编码盘、反射式光电检测、透射式光电检测和霍尔传感器检测。

经过对测速方案和其它学校方案的比较，本次设计中速度传感器采用的是OMRON公司生产的E6A2-CW3C型光电编码器，它由5-12V的直流供电。

速度传感器有三根引线，其中棕色线接VCC，蓝色线接地，黑色线为输出信号，需要上拉后接入单片机。车轮每前进一段距离时，速度传感器便产生一定数目的脉冲，单片机利用ECT 对脉冲进行计数，从而得到速度值。

通过测量给定时间里轮速脉冲信号的个数即可计算轮速：

$$V=\frac{2\pi rN}{Zt} \qquad (4.2)$$

式中，r为车轮半径，Z为齿圈齿数，N为频率信号输出脉冲个数，t为测量时间。编码器如图4.9。

图4.9　编码器实物图

4.6　电机驱动模块

如图所示，采用2个半桥智能功率驱动芯片BTS7960组合成一个全桥驱动器，驱动直流电机转动。BTS7960是电机驱动的大电流半桥集成芯片，他带有一个P沟道的高边MOSFET、一个N沟道的低边MOSFET和一个驱动IC。P沟道高边开关省去了电荷泵的需求，因而减少了电磁干扰。集成的驱动IC具有逻辑电平输入、电流诊断、斜率调节、死区时间产生和超温、过压、欠压、过流及短路保护功能。BTS7960的通态电阻典型值为16m欧姆，驱动电流可达43A，调节SR引脚外接电阻的大小可以调节MOS管导通和关断时间，具有防电磁干扰功能。IN引脚用于确定哪个MOSFET导通。当IN=1且INH=1时，高边MOSFET导通，输出高电平；当IN=0且INH=1时，低边MOSFET导通，输出低电平。通过对下桥臂开关管进行频率为25kHZ的脉宽调制（PWM）信号控制BTS7960的开关动作，实现对电机的正反向PWM驱动、反接制动、能耗制动等控制状态。

这块芯片开关频率可以达到25kHZ，可以很好地解决电机噪声大和发热的问题、同时驱动能力有了明显的提高，相应速度快。但是，电机变速时会使电源电压下降10%左右，控制器等其他电路容易产生掉电危险，从而使整个电路系统瘫痪。

图4.10为BTS7960的内部连接图：

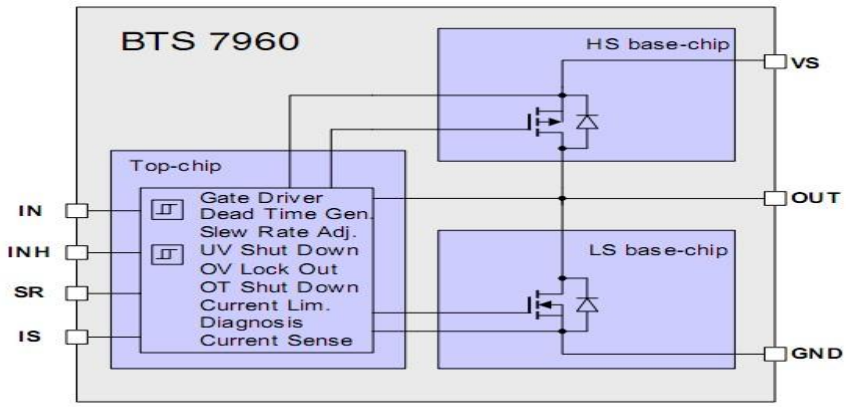

图4.10　BTS7960的内部连接图

图4.11为该芯片的封装图：

图4.11　BTS7960封装图

表4.2为BTS7960各管脚连接：

表4.2为BTS7960各管脚连接

管脚	名称	I/O	功能
1	GND	-	接地
2	IN	输入	输入（高或低有效）
3	INH	输入	抑制（低电平进入休眠状态）
4,8	OUT	输出	半桥功率输出
5	SR	输入	转换率调整（通过端口与地之间的电阻来调节）
6	IS	输入	电流检测与自我诊断
7	VS	-	供电

图4.12为驱动电路：

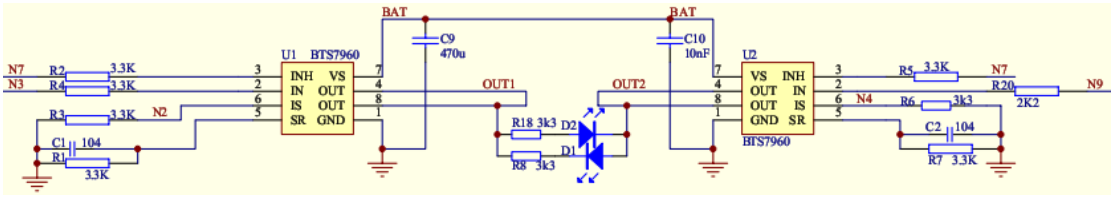

图4.12　驱动电路

图4.13为驱动模块实物图：

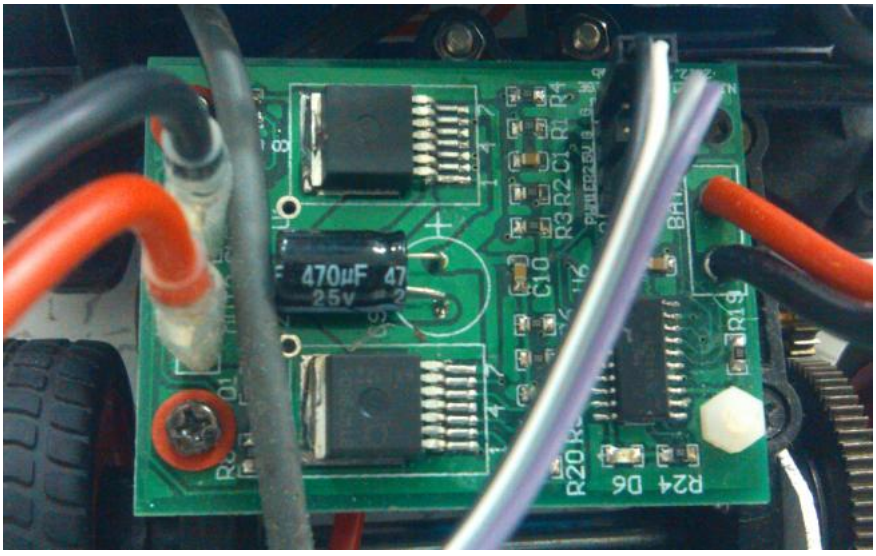

图4.13　驱动模块

4.7　小结

本章具体介绍了硬件系统各个模块的研究过程及设计方法，包括核心控制模块、电源模块、电机驱动模块、舵机驱动模块、传感器模块等。

第五章　图像处理和路况判断设计

5.1　图像采集方案设计

由于视觉传感器传送的数据量很大，而且传输的数据很快，单片机在有限的总线频率下无法采集所有的数据，所以必须对视觉传感器传送的信息进行优化处理。

由于视觉传感器的分辨率都较高，所以一幅图像扫描的行数和每行的点数都比较多。但对于飞思卡尔小车比赛，不需要把整幅图像的所有信息都采集进来。考虑到赛道窄道宽度为 45cm，导引黑线宽度为 2.5cm，所以只要采集能够分辨出导引黑线的行数和点数就可以了。同时，为了能够提高小车的速度，最好摄像头在一幅图像中能够分辨出单向弯和 S 弯，这样一旦检测到弯道信息，小车就有足够的时间做出入弯反应。因此本文有选择性地采集了 30 行，之所以有选择的选择扫描行，是因为摄像头的安装具有一定的俯角，图像产生了畸变。摄像头的架设示意图如图 5.1 所示。

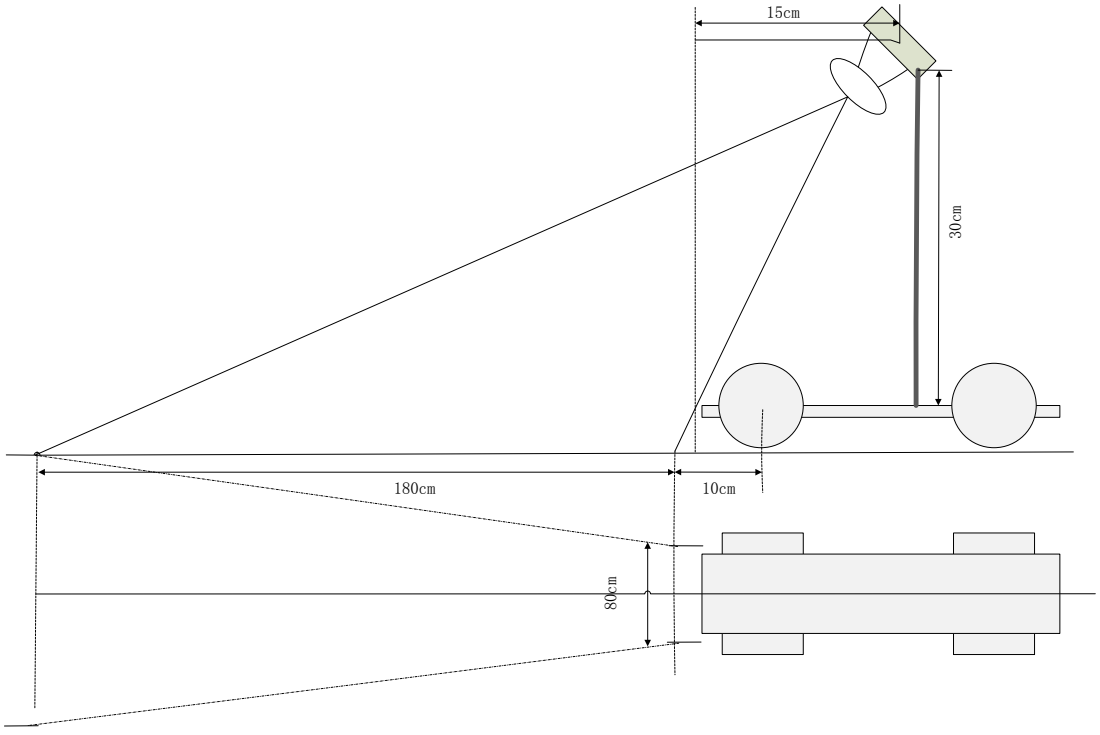

图 5.1　摄像头架构示意图

本届比赛引导线在赛道两边，要使摄像头在一行探到两条黑线，必须增加每行采集的点数。将总控单片机超频到 80M，根据此时普通 I/O 口的运行速率，发现每行最多可以采集 156 个点，将这 268 个点进行处理，已经满足了小车的赛道信息要求。

5.2 图像预处理算法设计

图像预处理的任务是从摄像头的每一行信息中提取出有效地黑色导引线，以便做后续分析。识别导引线常用的方法主要有阈值法和边缘检测法。

阈值法思想：通过实际调试，设定阈值，对采集到的 156*30 数组矩阵里的像素值与阈值比较。若像素值大于某阈值，则判定此像素对应的是白点；若像素值小于某阈值，则判定为黑点。

边缘检测法思想：设定阈值，不过不是用采集到的像素值直接与阈值比较，而是通过计算一行中相邻两个像素值作差，将此差值与设定的阈值比较。若大于阈值，则可判定检测到了黑白分界线。

经过大量调试发现，阈值法与边缘检测法效果相近，为计算方便，本文最终选择阈值法。由于系统采用的是数字式摄像头，免去了对图像噪声的处理。

由于摄像头安装高度以及倾斜角度造成采集的图像具有较大的梯形失真，主要表现为“近大远小”。所以远端的两个采样点之间的实际距离比近端的实际距离大，但单片机会认为它们是一样的，所以最好在软件中对梯形失真进行校正。横向无需校正，因为实际路径是取两引导线的中心。纵向通过有选择的选取行。具体做法是如图 5.2 在车前 180mm 前等间隔地贴上 30 条黑胶布，再用视频采集卡将图像截下来，通过测出每块黑胶布在图像中对应的行数来确定系统需要采集的行数。

图 5.2　选取要采集的行

5.3　赛道信息提取算法设计

利用图像滤波时记录的有效行检测到的两端黑线的位置，取两者的平均值作为实际路径。本文在赛道信息提取中主要使用了中心扩展法，具体实现过程如下：

单片机采集到一场图像后，首先根据近端第一行的数据计算黑色导引线的位置，并得到中心的位置。然后第二行以此中心位置为基准，分别向两边探测黑线位置，再得到第二行的中心位置，以此类推，每一行都以前一行的中心位置为基准，直到处理完整场信息。

这样做的优点是计算简单，不易丢线，安全可靠。图 5.3 为小车静止时，利用跟踪边缘检测法提取到的赛道信息。

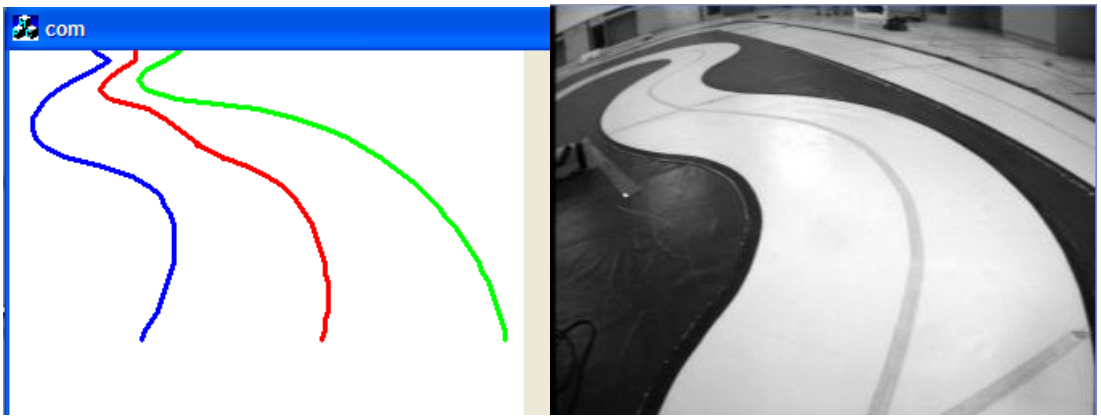

图 5.3　提取到的赛道信息（左图），实际赛道（右图）

5.4 路况判断算法设计

比赛中的赛道类型主要有直道、大 C 弯、小 C 弯、大 S 弯、小 S 弯以及十字交叉道。本文在路况识别上给出了两种识别方式，即位置偏移法和曲率计算法。

位置偏移法思想：将整幅图像的中心线位置相加，求其平均值。经过反复调试，得到赛道类别判定的阈值。本文通过平均值的大小和当前车速控制电机转速和舵机转角。

曲率计算法思想：利用提取的赛道上的三个点进行曲率计算，选点有讲究，首先要确保在有效行中选，其次三个点的分布距离要适当，不可太近或太远，否则计算得到的曲率精度不高。经过调试确定不同赛道类型的曲率阈值。本文利用赛道的判定类型，根据经验，在曲率大的地方适当减速；在曲率小的地方适当提速。

5.5 舵机控制算法设计

小车的方向控制是控制策略中的难点，因为小车运动系统是一个典型的惯性时滞系统。由于小车舵机用单片机产生的 PWM 波进行控制，而且使用的舵机是内部闭环。所以本文采用的舵机控制方略为小车在不同位置时，给定舵机相应的转向控制。舵机的转向控制策略有分为三部分，即直道舵机转向控制策略、普通弯道舵机转向策略以及大（小）S 弯舵机转向策略。

5.6 电机控制算法设计

为了能使小车的速度得到稳定的控制，电机的控制采用 PID 控制算法。

PID 控制算法思路：该算法结构简单、鲁棒性强，是自动控制领域中广泛采用的一种控制方法。常规的 PID 控制算法有三种控制参数组成，即比例、积分和微分。比例系数K_p对系统的稳定性、超调量和相应速度起着主要影响。增大K_p可以提高响应速度，但会影响系统的稳定性。一般来说，当偏差较大时可以增

大K_p，当偏差较小时可以减小K_p。积分可以提高系统的型别（有无误差），有利于系统稳态性能的提高。微分可以克服大惯性时间参数的影响，对动态调节过程影响很大，增大K_d有利于减小超调，但调节时间增大。

PID 控制算法的公式：

$$U = K_p e + K_i \int_0^t e dt + K_d e \tag{5.1}$$

PID 参数的选取在稳定性和响应速度之间存在矛盾，为了获取满意的系统性能，在控制中应根据系统的动态特征，采取变 P 控制，以增强系统的适应性。对于小车系统而言，利用 P 分量就已经得到很好的控制效果了。

本文的电机控制算法采用的是增量式 PID 算法，公式如下所示：

增量式 PID 控制算法公式：

$$\Delta U_n = U_n - U_{n-1}$$

$$= K_p * (e_n - e_{n-1}) + K_i * e_n + K_d * (e_n - 2 * e_{n-1} + e_{n-2}) \tag{5.2}$$

离散化后的采样时间为 20ms。

电机闭环控制原理如图 5.4 所示：

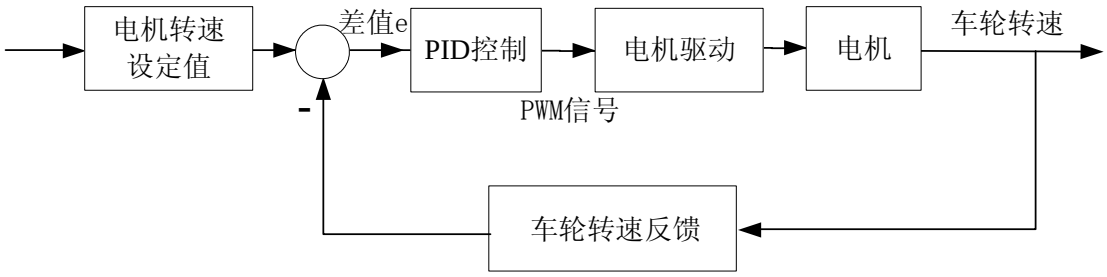

图 5.4　电机闭环控制

5.7　小结

本章具体介绍了图像处理的各个步骤，包括：图像预处理算法、赛道信息提取算法、路况判断以及舵机和电机的控制算法。

第六章　系统调试

6.1　软件调试平台

在对程序进行开发和软硬件联调的过程中需要一整套的软件开发与调试工具。程序的开发是在组委会提供的CodeWarrior IDE下进行的，包括源程序的编写、编译和链接，并最终生成可执行文件。CodeWarrior for S12 是面向以HC1和S12为CPU的单片机嵌入式应用开发软件包。包括集成开发环境IDE、处理器专家库、全芯片仿真、可视化参数显示工具、项目工程管理器、C交叉编译器、汇编器、链接器以及调试器。使用BDM来下载程序，把编译好的程序下载到单片机里运行。

赛车的硬件开发工具主要为Protel 99SE，通过该软件来完成电路原理图的绘制以及PCB板制作。

Code Warrior开发软件使用界面如图6.1所示。

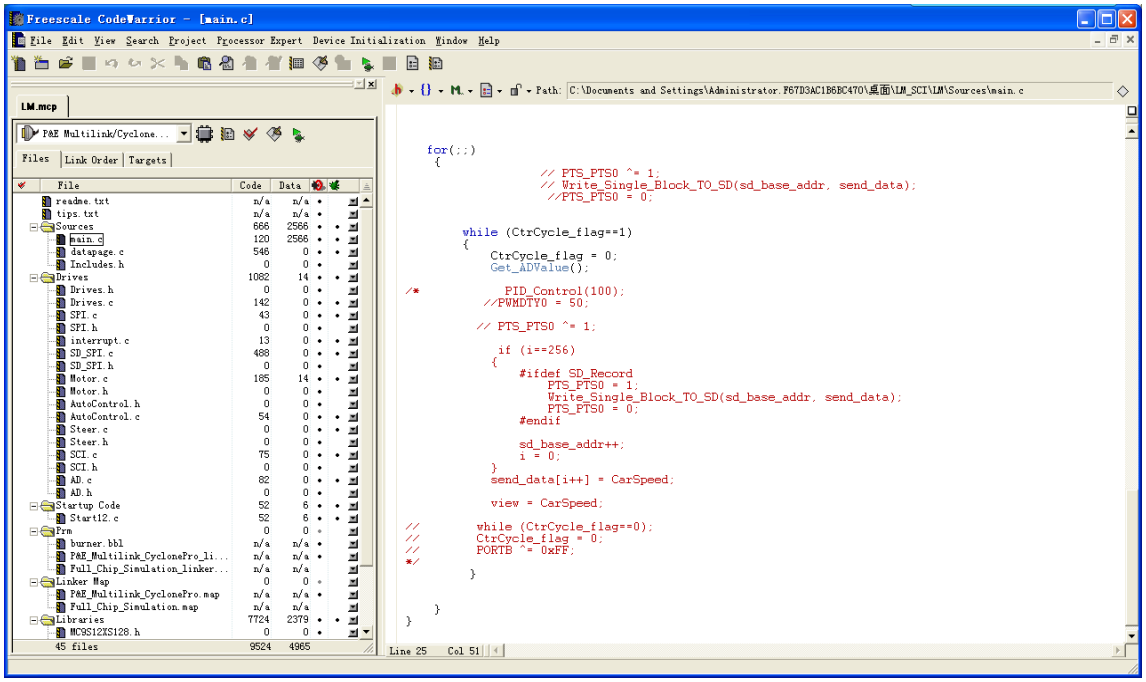

图6.1　CodeWarrior使用界面

Code Warrior是面向以HC12和S12为CPU的单片机嵌入式应用开发的软件包，

包括集成开发环境IDE、处理器专家库、全芯片仿真、可视化参数显示工具、项目工程管理器、C交叉编译器、汇编器、链接器以及调试器等。在CodeWarrior软件中可以使用汇编语言或C语言，以及两种语言的混合编程。

用户可在新建工程时将芯片的类库添加到集成环境开发环境中，工程文件一旦生成就是一个最小系统，用户无需再进行繁琐的初始化操作，就能直接在工程中添加所需的程序代码。

在原程序编译连接通过后，就可以进行程序的下载了。下载前先将单片机已存在的部分擦除，擦除界面如图6.2所示，擦除完了以后点击Load，将bin文件夹下生成的后缀为.abs的文件打开，就可以完成下载。下载使用BDM下载器进行上位机和下位机的互联，BDM如图6.3所示。

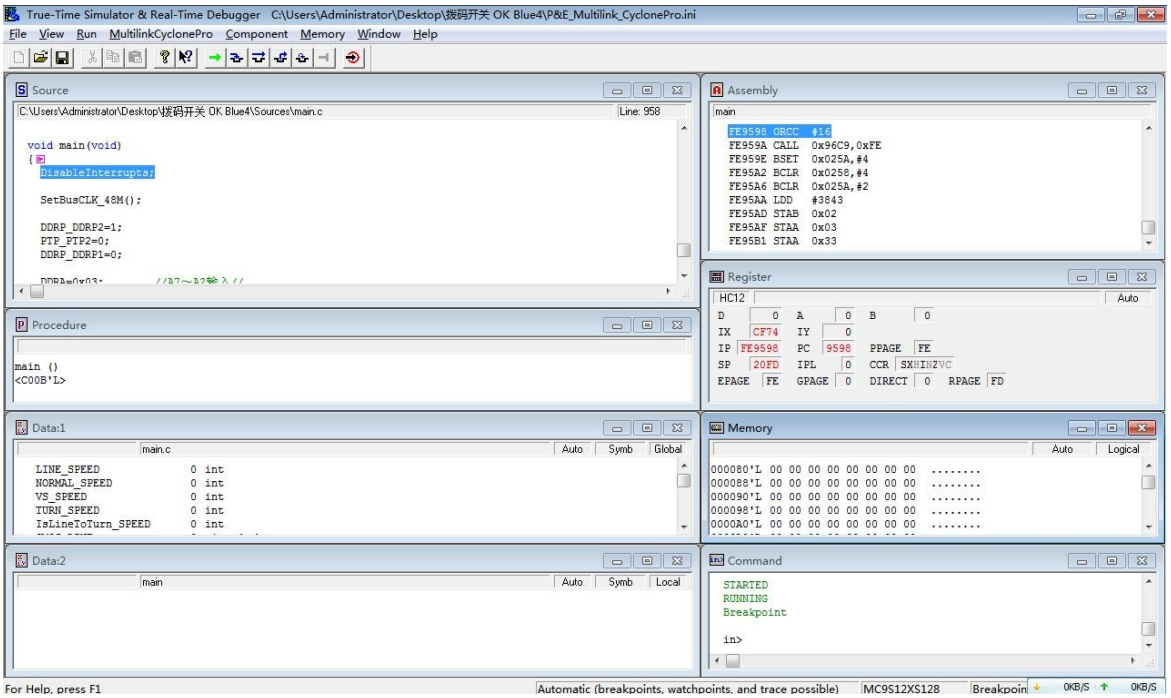

图6.2 程序擦除烧写界面

图6.3　BDM下载器

6.2　硬件调试

6.2.1　舵机模块调试

首先，我们使用电子调速器配合遥控器对车模进行了测试。通过遥控车模行进和转弯，观察舵机是否能流畅完成遥控器发出的指令。经过测试，能很好完成各种转弯，并且直线不会跑歪。

然后将舵机与电路板进行联调，根据舵机的接口特性，通过子板向母板的舵机接口输出不同占空比的 PWM 信号，检测舵机的偏转角度。使占空比增加或减少相同的值，观察舵机向两个方向转过的角度是否基本相同。根据测量，当输出 PWM 数值为 1552 时，小车可以稳定的沿直线行驶；输出 PWM 数值为 1692 时，舵机往左最大，输出 PWM 数值为 1412 时，舵机往右最大。

6.2.2　摄像头模块调试

摄像头的焦距直接影响到采到图像的清晰度，即路径识别的好坏。在摄像头接入系统之前要先调整摄像头的焦距，我们采用的单板机出厂时的焦距已经固定，清晰度已经调整合适，因此不需要再进行焦距的调整。系统将图像通过串口发送给上位机显示，以此检测图像的正确性，如图 6.4：

图 6.4　上位机上显示的图像

单片机对摄像头采集的图像进行处理，将两条黑线的位置发送到上位机。如图 6.5，蓝线为左边线，绿线为右边线，红线为中心线，黑线为起点到终点的连线。

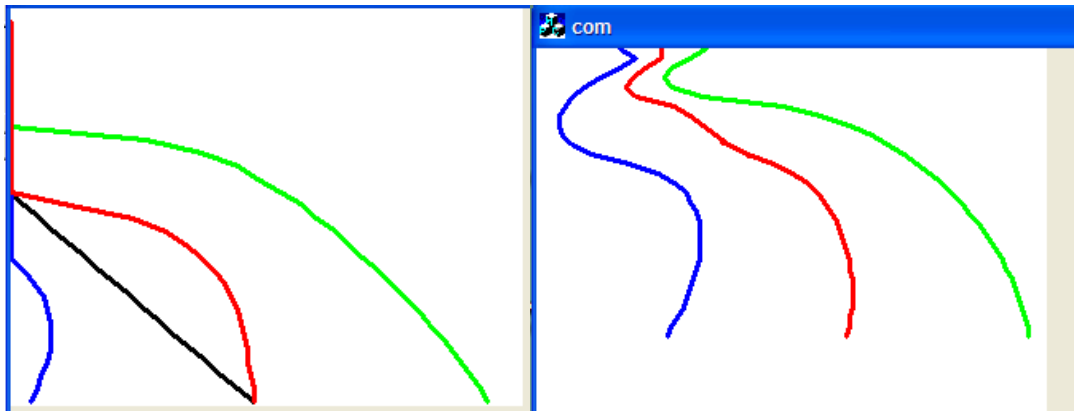

图 6.5　经处理后的各种路况图像

6.2.3　编码器调试

根据编码器手册定义，确认编码器的接口正确。由于编码器输出脉冲数和距离成正比，因此，只要比较脉冲比和距离比是否一致来判断编码器的性能。根据实际测试，让小车推行 1 米距离后停止，测出编码器输出大概 3800 个脉冲数，然后再推行 5 米距离，测出脉冲数约为 19000 ，两比例一致，都为 5，所以编码器工作正常。

6.3　本章小结

本章介绍了智能车系统的开发环境以及各功能模块的调试过程。

参考文献

[1] 邵贝贝.单片机嵌入式应用的在线开发方法[M].北京：清华大学出版社，2007.
[2] 余志生.汽车理论[M].北京：清华大学出版社，2009
[3] 谭建成.新编电机控制专用集成电路与应用.北京：机械工业出版社，2005
[4] 王威等.HCS12 微控制器原理及应用[M].北京：北京航空航天大学出版社，2007.
[5] 卓晴，黄开胜，邵贝贝等.学做智能汽车[M].北京：北京航空航天大学出版社，2007.
[6] 吴红星.电机驱动与控制专用集成电路应用[M].北京：中国电力出版社，2006.
[7] 江思敏，唐广芝.PCB 和电磁兼容性设计[M].北京：机械工业出版社，2008.
[8] 王庆友，孙学珠.CCD 应用技术[M].天津：天津大学出版社，1993.
[9] 姜雪松，陈绮，许灵军，范博.印制电路板设计[M].北京：机械工业出版社，2006.
[10] 姜培安.印制电路板的可制造性设计[M].北京：中国电力出版社，2007.
[11] Andrea Goldsmith. Wireless Communications[M]. Cambrigde University Press,2005 .
[12] 杨宗德.Protel DXP 电路设计制板 100 例[M].北京：人民邮电出版社，2005.
[13] 陈伯时.电力拖动自动控制系统——运动控制系统[M].北京：机械工业出版社，2008.
[14] 曾峰，巩海波，曾波.印制电路板（PCB）设计与制作[M].北京：电子工业出版社，2005.
[15] 童诗白，华成英.模拟电子技术基础[M].北京：高等教育出版社，2004.
[16] 熊有伦.机器人技术基础[M].武汉：华中理工大学出版社，1997.
[17] 杜刚.电路设计与制板：Protel 应用教程[M].北京：清华大学出版社，2006.
[18] 聂荣等.实例解析 PCB 设计技巧[M].北京：机械工业出版社，2006.
[19] 肖景和.数字集成电路应用精粹[M]. 北京：人民邮电出版社，2002.
[20] （日）米本和也.CCD/CMOS 图像传感器基础与应用[M].北京：科学出版社，2006.
[21] 敖荣庆，袁坤.伺服系统[M].北京：航空工业出版社，2006.
[22] 位元文化.从 C++,面向对象到窗口程序设计[M].武汉：华中理工大学出版社，1999.
[23] 周立功等.ARM 微控制器基础与实战[M].北京：北京航空航天大学出版社，2006.
[24] 杨国田，白焰. 摩托罗拉 68HC12 系列微控制器原理、应用与开发技术[M].北京：中国电力出版社，2003.
[25] 康华光.电子技术基础模拟部分（第四版）[M].北京：高等教育出版社，1996.
[26] 郭烈.高速智能汽车电器与控制系统设计开发[D].长春：吉林大学，2003.
[27] 赵玲.基于视觉和超声传感器融合的移动机器人导航系统研究[D].武汉：武汉理工大学，2007.
[28] 胡杰.基于 16 位单片机 MC9S12DG128 智能模型车系统开发研究[D].武汉：武汉理工大学，2008.

[29] 徐建洪.基于S3C2410的嵌入式智能汽车控制系统设计[D].南京:南京理工大学,2008.
[30] 谢恒义.边防哨所重要路口无人值守自动监控系统的研究[D].成都：成都理工大学，2008.
[31] 陈宏杰.消防机器人无线视频传输系y统研究[D].长春：吉林大学,2007.
[32] 王朝盛.基于16位单片机MC9S12DG128B智能汽车系统的设计[D].天津：天津工业大学，2007.
[33] 伍舜喜.基于激光雷达的智能汽车定位技术研究[D].上海：上海交通大学，2008.
[34] 苏鑫.小型智能汽车自动驾驶系统设计与实现[D].西安：西安理工大学，2008.
[35] 田海波.一种微型智能汽车的结构设计与性能分析[D].西安：西北工业大学，2006.
[36] 王明铭.智能模型汽车控制系统硬件设计[D].南京：南京理工大学，2009.
[37] 梁炎昌. 基于嵌入式系统的图像采集技术研究[D].西安:西安电子科技大学,2007.
[38] 严正国.井下电视成像测并图像压缩技术研究[D].西安：西安电子科技大学，2005.
[39] 林文斌.实时无线多媒体传输系统中视频压缩编码的设计与实现研究[D].杭州：浙江工业大学,2004.
[40] 何奇文，彭建盛，周东.基于红外反射式传感器智能汽车系统的设计[J].高师理科学刊，2008，28（3）：45-48.
[41] A.Broggi.The ARGO Autonomous Vehicle’s Vision and Control Systems. International Journal of Intelligent Control and Systems, 1999, Vol.3,No.4:409-441.
[42] 邱丹，王东，高振东.直流电机PWM闭环调速系统[J].青岛大学学报，2000，15（1）：10-12.
[43] Myung-Wook Park,Young-Jin Son,Jung-Ha Kim.Design of the Real Time Control System for Controlling Unmanned Vehicle.International Conference on Control, Automation and Systems, Oct 2007,pp.1234-1237.
[44] 徐友春，王荣本.世界智能车辆近况综述[J].车辆工程，2001，23（5）：189-195.
[45] 李峰.智能交通系统在国外的发展趋势[J].国外公路，1999，19（1）：1-5.
[46] 王诗恩，何仁.电动汽车的关键技术[J].江苏理工大学学报，1996，17（5）：35-40.
[47] 李果. 汽车自动控制系统设计与研究[D].长沙：国防科技大学，1994.
[48] 项小峰. 交通信息采集设备视频压缩模块的设计与实现[D].杭州：浙江大学，2004.
[49] 马雷,王荣本,赵东标. 智能车辆导航控制器参数选取与鲁棒性分析[J]. 机械科学与技术, 2005,24(2):146-150.
[50] 周红妮. 车辆稳定性控制方法与策略的比较研究[D]. 武汉:武汉科技大学, 2006.
[51] 吕广明. 自动引导车轨迹偏差的智能控制[J]. 哈尔滨工业大学学报,2003，35(12):1465-1467.
[52] 程洪. 智能车视觉导航算法及其系统实现的研究[D]. 西安:西安交通大学,2003.
[53] 王荣本，张荣辉，游峰，储江伟，金立生. 智能车辆弧线跟踪控制算法[J]. 吉林大学学报, 2006, 36(5):731-735.

[54] 郭孔辉. 汽车操纵稳定性[M]. 吉林：吉林人民出版社，1983.
[55] 刘涛. 两种驾驶员方向控制模型的比较与研究[D]. 吉林: 吉林大学, 2007.
[56] 高振海，管欣，李谦，郭孔辉. 基于预瞄跟随理论的驾驶员跟随汽车目标速度的控制模型[J]. 吉林大学学报，2002, 32(1):1－4.
[57] 安向京. 自动驾驶汽车视觉导航系统研究——理论、方法和系统实现[D]. 长沙:国防科技大学，2001.
[58] 毕雁冰. 高速汽车车道偏离预警系统可行区域感知算法研究[D]. 吉林:吉林大学，2006.
[59] 陈无畏，施文武，王启瑞，何世娣. 基于动力学模型的自动引导车智能导航控制研究[J]. 农业机械学报，2003,34(4):90-93.
[60] 王荣本，李兵，徐友春，李斌. 基于视觉的智能车自主导航最优控制器设计[J]. 汽车工程，2001, 23(2):97-100.
[61] 彭湘德. 汽车电子化技术的发展[J]. 制造业自动化，1990，1.
[62] 智能汽车. http://baike.baidu.com/view/263534.htm, 2010-5-22.

附录一：控制系统核心代码

```
#include <hidef.h>      /* common defines and macros */
#include <MC9S12XS128.h>     /* derivative information */
#include "PE_Types.h"
#include "init.h"
#include "PID.h"
#pragma LINK_INFO DERIVATIVE "mc9s12xs128"

bool field_over=0;
unsigned char camdata[31][156]={0};
unsigned char *p=camdata[0];
unsigned char hangnum[30]={
                            14,31,46,60,74,86,
                            98,108,118,128,136,145,
                            153,160,167,173,180,186,
                            191,197,202,207,211,216,
                            220,224,228,232,236,239

                            };
unsigned char digital[10]={0xc0,0xf9,0x64,0x70,0x59,0x52,0x42,0xf8,0x40,0x50};
unsigned char *q=hangnum;

unsigned char left[31]={0};
unsigned char right[31]={0};
unsigned char center[31]={0};
int width[36]={0};

short Max,Min;
unsigned char Maxhang,Minhang;
float livek,Sumlivek;
unsigned char liveline[31];
char Crve_count=0;

unsigned char nihe_count=0;

short left_k1[30]={0};
```

```
short right_k1[30]={0};

bool breakflag=0;
bool stop_flag=0;
unsigned char final_stop=0;
bool stopflag_en=0;

unsigned int dingshi_stop=0;
unsigned int dingshi_start=0;
char yuan_point=30,jin_point=30,zhong_point=30;

uchar RoadType = 1;
char Direct=0;

unsigned char hang=0;
unsigned char href_num=0;

SPD_PID SpdPID=SpdPIDDefaults;
STR_PID StrPID=StrPIDDefaults;

int SPD_Stright,SPD_Crve,SPD_BigS,SPD_InCrve,SPD_SmallS;
unsigned char blackhold;


#define  Stright             1
#define  InCrve              2
#define  Crve                3
#define  SmallS              4
#define  BigS                5
#define  Out                 6
#define  Polluted            7

#define Left -1
#define Right 1
#define Zhong 0

void delayms(int ms)
{
   int ii,jj;
   ms=ms*4;
```

```
   for(ii=0;ii<ms;ii++)
     for(jj=0;jj<3338;jj++);    //80MHz--1ms
}

float Abs(float a)
{
  if(a<0) a=-a;
  return (float)a;
}

void GetBlackDots01(void)
{
  int i;
  uchar *p;
  p=&camdata[0][center[0]];
  for(i=center[0];*(p--)>blackhold&&i>0;i--) {}
  if(i==center[0])  RoadType=Out;
  left[0]=i;
  p=&camdata[0][center[0]];
  for(i=center[0];*(p++)>blackhold&&i<155;i++) {}
  if(i==center[0])  RoadType=Out;
  right[0]=i;
  center[0]=(left[0]+right[0])/2;
  width[0]=right[0]-left[0];
}

void GetBlackDots1_291(void)
{
  int i;
  uchar *p;
  char temphang;
  int tempcenter;
  breakflag=0;

  Direct=Zhong;
  nihe_count=0;
  for(temphang=1;temphang<=29;temphang++)
  {
    p=&camdata[temphang][center[temphang-1]];
    for(i=center[temphang-1];*(p--)>blackhold&&i>0;i--) {}
```

```
    left[temphang]=i;
    p=&camdata[temphang][center[temphang-1]];
    for(i=center[temphang-1];*(p++)>blackhold&&i<155;i++) {}
    right[temphang]=i;
    center[temphang]=(left[temphang]+right[temphang])/2;
    width[temphang]=right[temphang]-left[temphang];

    left_k1[temphang]=left[temphang]-left[temphang-1];
    right_k1[temphang]=right[temphang]-right[temphang-1];


if((left_k1[temphang]<-5&&left_k1[temphang-1]>0)||(right_k1[temphang]>5&&right
_k1[temphang-1]<0)&&stop_flag==0)
    {
      center[temphang]=center[temphang-1];
      left[temphang]=left[temphang-1];
      left_k1[temphang]=1;
      right[temphang]=right[temphang-1];
      right_k1[temphang]=-1;
      nihe_count++;
    }
    else
if(RoadType==Stright&&(left_k1[temphang]>15||right_k1[temphang]<-15)&&(left_k1
[temphang-1]<3&&right_k1[temphang-1]>-3)

&&(left[temphang-1]>5&&right[temphang-1]<150)&&Abs(center[temphang-1]-78)<30&&
nihe_count==0)
    {
      if(temphang<20||dingshi_start<100)
      {
        if(dingshi_start==100&&stopflag_en==1)
        {
          stop_flag=1;
          dingshi_stop=0;
        }
        center[temphang]=center[temphang-1];
        left[temphang]=left[temphang-1];
        left_k1[temphang]=1;
        right[temphang]=right[temphang-1];
        right_k1[temphang]=-1;
```

```
      }
      else ;

    }
    else if(stop_flag==0&&width[temphang]<5)
    {
      yuan_point=temphang;
      if(center[temphang]<78) Direct=Left;
      else if(center[temphang]>78) Direct=Right;
      breakflag=1;
      break;
    }
  }
  if(breakflag==1)
  {

    if(Direct==Left)
    {
      for(;temphang>0;temphang--)
      {

if((left[temphang]==0&&left[temphang-1]!=0)||(width[temphang]>width[temphang-1
]&&left[temphang]!=0))
       {
         temphang--;
         break;
       }
      }
      jin_point=temphang+1;
      if(yuan_point-jin_point>15)
        jin_point=yuan_point-13;
      zhong_point=(jin_point+yuan_point)/2;
      for(temphang=29;temphang>=yuan_point;temphang--)
      {
        left[temphang]=0;
        center[temphang]=0;
        right[temphang]=0;
      }
      for(;temphang>=zhong_point;temphang--)
      {
```

```
        left[temphang]=0;
        center[temphang]=0;
      }
      for(;temphang>=jin_point;temphang--)
      {
        left[temphang]=0;
      }

      for(temphang=zhong_point-1;temphang>=jin_point;temphang--)
      {
        center[temphang]=(center[temphang-1]+center[temphang+1])/2;
      }

    }
    else if(Direct==Right)
    {
      for(;temphang>0;temphang--)
      {

if((right[temphang]==155&&right[temphang-1]!=155)||(width[temphang]>width[temp
hang-1]&&right[temphang]!=155))
        {
         temphang--;
          break;
        }
      }
      jin_point=temphang+1;
      if(yuan_point-jin_point>15)
        jin_point=yuan_point-13;
      zhong_point=(jin_point+yuan_point)/2;
      for(temphang=29;temphang>=yuan_point;temphang--)
      {
        left[temphang]=155;
        center[temphang]=155;
        right[temphang]=155;
      }
      for(;temphang>=zhong_point;temphang--)
      {
        right[temphang]=155;
        center[temphang]=155;
```

```
      }
      for(;temphang>=jin_point;temphang--)
      {

        right[temphang]=155;
      }

      for(temphang=zhong_point-1;temphang>=jin_point;temphang--)
      {
        center[temphang]=(center[temphang-1]+center[temphang+1])/2;
      }
    }
  }
  else
  {
    yuan_point=29;
    zhong_point=29;
    jin_point=29;
    Direct=Zhong;
  }
}

void SmoothLine(void)
{
  uchar i;
  Max=0;
  Min=155;
  for(i=1;i<zhong_point;i++)
  {
    center[i]=(center[i-1]+center[i+1])/2;
  }
}

void get_liveline(void)
{
  unsigned char i;
  int tempError;
  unsigned char tempMaxhang,tempMinhang;
  short tempMax,tempMin;
  livek=(center[zhong_point]-center[0])/(float)zhong_point;
```

```
Sumlivek=0;
liveline[0]=center[0];
Sumlivek=center[0];
Max=3;
Min=-3;
Maxhang=0;
Minhang=0;
for(i=1;i<zhong_point;i++)
{
  Sumlivek+=livek;
  liveline[i]=Sumlivek;
  tempError=center[i]-liveline[i];
  if(tempError>Max)
  {
    Max=tempError;
    Maxhang=i;
  }
  if(tempError<Min)
  {
    Min=tempError;
    Minhang=i;
  }
}
for(;i<30;i++)
{
  liveline[i]=center[i];
}

if((Maxhang!=0&&Minhang==0)||(Maxhang==0&&Minhang!=0))
{
  tempMax=0;
  tempMin=155;
  for(i=0;i<=zhong_point;i++)
  {
    if(center[i]>tempMax)
    {
     tempMax=center[i];
     tempMaxhang=i;
    }
    if(center[i]<tempMin)
```

```
      {
       tempMin=center[i];
       tempMinhang=i;
      }
    }
    if(tempMaxhang==zhong_point)  tempMaxhang=0;
    if(tempMinhang==zhong_point)  tempMinhang=0;

    if(Abs(center[tempMaxhang]-center[0])>10)
    {
     Max=center[tempMaxhang]-liveline[tempMaxhang];
     Maxhang=tempMaxhang;
    }
    if(Abs(center[tempMinhang]-center[0])>10)
    {
     Min=center[tempMinhang]-liveline[tempMinhang];
     Minhang=tempMinhang;
    }
  }

}


void Judge_state(void)
{
  unsigned char temphang;
  float k1,k2;
  if(Maxhang!=0&&Minhang==0)
  {
    temphang=Maxhang;
    k1=(center[temphang]-center[0])/(float)temphang;
    k2=(center[zhong_point]-center[temphang])/(float)(zhong_point-temphang);
  }
  else if(Minhang!=0&&Maxhang==0)
  {
    temphang=Minhang;
    k1=(center[temphang]-center[0])/(float)temphang;
    k2=(center[zhong_point]-center[temphang])/(float)(zhong_point-temphang);

  }
```

```
  else temphang=255;

  if(Max-Min<15&&breakflag==0)
    RoadType=Stright;
  else
  {
    if(temphang==255)
    {
      if(Abs(livek)<2)
        RoadType=SmallS;
      else
        RoadType=BigS;
    }
    else
    {
      if((Abs(k1)<=0.7||Abs(k2-livek)<=1)&&breakflag==0)
        RoadType=Stright;
      else if((k1<0.5&&k2<-1)||(k1>-0.5&&k2>1))

RoadType=(left_k1[jin_point]<-9||right_k1[jin_point]>9||yuan_point<16)?InCrve:
Crve;
      else if((k1>1&&k2<-1)||(k1<-1&&k2>1))
        RoadType=BigS;
      else
        RoadType=SmallS;
    }
  }
}

// ====================================================================
//速度控制模块
void  SpeedPID(void)
{
 Crve_count++;
 switch(RoadType)
 {
  case Stright:
  {
    SpdPID.Kp = 20;
    SpdPID.Ki = 4;
```

```
    SpdPID.Kd = 0;
    SpdPID.Ref=(width[29]<45)?SPD_Stright:100;
    //Crve_count=0;
    break;
  }
  case BigS:
  {
    SpdPID.Kp = 20;
    SpdPID.Ki = 1;
    SpdPID.Kd = 0;
    SpdPID.Ref=SPD_BigS;
    //Crve_count=0;
    break;
  }
  case SmallS:
  {
    SpdPID.Kp = 20;
    SpdPID.Ki = 4;
    SpdPID.Kd = 0;
    SpdPID.Ref=SPD_SmallS;
    //Crve_count=0;
    break;
  }
  case InCrve:
  {
    SpdPID.Kp = 20;
    SpdPID.Ki = 4;
    SpdPID.Kd = 0;

SpdPID.Ref=(left_k1[jin_point]<-9||right_k1[jin_point]>9)?70:SPD_InCrve;//70;
    Crve_count=0;
    break;
  }
  case Crve:
  {
    SpdPID.Kp = 20;
    SpdPID.Ki = 1;
    SpdPID.Kd = 0;
    SpdPID.Ref=(Crve_count>15)?SPD_Crve:70;
    break;
```

```
  }
  default: break;
 }
 if(Crve_count>20) Crve_count=20;
 SpdPID.Fdb=PACNT;
 PACNT=0;
 if(dingshi_stop==5)
 {
  SpdPID.Kp = 20;
  SpdPID.Ki = 1;
  SpdPID.Kd = 0;
  SpdPID.Ref=0;
  Crve_count=0;
  SpdPID.Calc(&SpdPID);
  if(SpdPID.Fdb<5)
    SpdPID.U=-90;
  if(SpdPID.Fdb==0)
    final_stop++;
 }
 else
  SpdPID.Calc(&SpdPID);

 if(final_stop>10)
 {
  final_stop=11;
  SpdPID.U=0;
 }
 PWMDTY23=1000+SpdPID.U;
 PWMDTY67=1000-SpdPID.U;
}



//====================================================================
//转角控制模块
void DirectPID(void)
{
  unsigned char i;
  unsigned int tempError=0;
  switch(RoadType)
```

```
{
  case Stright:
  {
    StrPID.Kp = 2;
    StrPID.Ki = 0;
    StrPID.Kd = 8;
    StrPID.Ref=78;
    StrPID.steerhang=15;
    StrPID.Fdb=center[StrPID.steerhang];
    break;
  }
  case BigS:
  {
    StrPID.Kp = 3;
    StrPID.Ki = 0;
    StrPID.Kd = 1;
    StrPID.Ref=78;
    StrPID.steerhang=13;
    StrPID.Fdb=center[StrPID.steerhang];
    break;
  }
  case SmallS:
  {
    StrPID.Kp = 1.5;
    StrPID.Ki = 0;
    StrPID.Kd = 8;
    StrPID.Ref=78;
    StrPID.steerhang=12;
    StrPID.Fdb=center[StrPID.steerhang];
    break;
  }
  case InCrve:
  {
    StrPID.Kp = 3;//5
    StrPID.Ki = 0;
    StrPID.Kd = 6;
    StrPID.Ref=78;
    StrPID.steerhang=6;//8
    StrPID.Fdb=center[StrPID.steerhang];
    break;
```

```
    }
    case Crve:
    {
      StrPID.Kp = 3.2;    //3
      StrPID.Ki = 0;
      StrPID.Kd = 7;    //8
      StrPID.Ref=78;
      StrPID.steerhang=8; //8
      StrPID.Fdb=center[StrPID.steerhang];
      break;
    }
    default: break;
  }


  StrPID.Calc( &StrPID );
  PWMDTY45=1552+StrPID.U;   //1545
}
//====================================================================
void key_control(void)
{
  switch(PORTB&0b00000111)   //Stright
  {
    case 0b00000000:{SPD_Stright=100;SPD_Crve=90;break;}
    case 0b00000001:{SPD_Stright=120;SPD_Crve=110;break;}
    case 0b00000010:{SPD_Stright=130;SPD_Crve=120;break;}
    case 0b00000011:{SPD_Stright=140;SPD_Crve=120;break;}
    case 0b00000100:{SPD_Stright=145;SPD_Crve=120;break;}
    case 0b00000101:{SPD_Stright=150;SPD_Crve=120;break;}
    case 0b00000110:{SPD_Stright=155;SPD_Crve=120;break;}
    case 0b00000111:{SPD_Stright=160;SPD_Crve=140;break;} //160//130
    default: break;
  }
  switch(PORTB&0b00011000)   //InCrve
  {
    case 0b00000000:{SPD_InCrve=60;SPD_BigS=70;break;}
    case 0b00001000:{SPD_InCrve=70;SPD_BigS=80;break;}
    case 0b00010000:{SPD_InCrve=80;SPD_BigS=90;break;}
    case 0b00011000:{SPD_InCrve=110;SPD_BigS=120;break;} //90//100
    default: break;
```

```
  }
  switch(PORTB&0b00100000)   //SmallS
  {
    case 0b00000000:{SPD_SmallS=SPD_BigS;break;}
    case 0b00100000:{SPD_SmallS=SPD_Stright;break;}
    default: break;
  }
  switch(PORTB&0b01000000)   //stopflag
  {
    case 0b00000000:{stopflag_en=0;break;}
    case 0b01000000:{stopflag_en=1;break;}
    default: break;
  }
  switch(PORTB&0b10000000)   //yuzhi
  {
    case 0b00000000:{blackhold=100;break;}
    case 0b10000000:{blackhold=140;break;}
    default: break;
  }

}

void main(void)
{
  unsigned char i=0;
  INIT_ALL();

  PIFP = 1U;                           // Clear flag
  PIEP_PIEP0 = 1U;                     // Enable interrupt

  center[0]=78;
  DDRJ=0x01;
  DDRB=0x00;

  SPD_Stright=100;
  SPD_Crve=90;
  SPD_BigS=70;
  SPD_InCrve=60;
  SPD_SmallS=70;
  stopflag_en=1;
```

```
blackhold=140;
key_control();

delayms(1000);
field_over=0;

for(;;)
{
  if(field_over==1)
  {
      GetBlackDots01();
      GetBlackDots1_291();
      SmoothLine();
      get_liveline();
      Judge_state();
      PORTK=digital[RoadType];

      SpeedPID();
      liveline[30]=250;
      left[30]=0;
      center[30]=78;
      right[30]=155;

      switch (i)
      {
        case 0: {send(left,31);i++;break;}
        case 1: {send(center,31);i++;break;}
        case 2: {send(right,31);i++;break;}
        case 3: {send(liveline,31);i++;break;}
        case 4: {i=0;break;}
        default:break;
      }

      DirectPID();
      if(RoadType==Out)
      {
        if(Direct==Right)  PWMDTY45=1552+140;
        if(Direct==Left)   PWMDTY45=1552-140;
      }
      if(dingshi_stop!=5&&stop_flag==1)
```

```
          dingshi_stop++;
        if(dingshi_start!=100)
          dingshi_start++;
        field_over=0;
    }
  }
}

#pragma CODE_SEG __NEAR_SEG NON_BANKED
void interrupt 56 VSYNC(void)
{
  PIFP =  1U;                           /* Clear flag */

  hang=0;
  q=hangnum;
  href_num=0;
  field_over=1;

  //PIFH=1;
  //PIEH_PIEH0 = 1U; //打开行中断
}
#pragma CODE_SEG __NEAR_SEG NON_BANKED
void interrupt 25 HREF(void)
{

  PIFH = 1U;                            /* Clear flag */

  if(href_num==*q)
  {
    p=camdata[hang++];
    q++;

    while(PTH_PTH0)
    {
      while(!PORTE_PE5){}
      *(p++)=PORTA;
    }
  }
  href_num++;
}
```