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：西北工业大学

队伍名称：勇毅队

队　　员：程　立

王　全

刘沛西

带队教师：曲仕茹、陈小锋

2012 年 8 月 15 日

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

智能汽车综合应用环境感知、模式识别、控制决策、计算机以及电子等技术，使汽车具有自动识别道路，并能完成自主行驶等功能。

本技术报告以“飞思卡尔”杯智能汽车竞赛为背景，详细介绍了制作具备自主识别路径，并快速、稳定行驶在跑道上的智能车的设计方案及其实现技术。本技术报告首先介绍了系统的总体设计思路，并详细阐述了系统硬件、软件以及路径识别与控制相关算法的设计与实现，同时对车模的机械调整和系统调试进行了说明，最后系统地总结了智能车设计、制作和调试过程中遇到的各种问题。

本智能车控制系统采用 Freescale 16 位微控制器作为核心控制单元，通过采集并处理摄像头信号来控制转向，同时接收车速传感器的反馈信号对车速加以控制。为了提高智能车控制性能，我们对系统进行了创造性的优化：首先，在硬件上采用单片机 MC9S12XS128 作为核心控制器，并设计了数字式和模拟式摄像头兼容的路径识别方案；同时，使用两片 BTS7960 来驱动电机，很好的解决了一片驱动芯片驱动电流小、温度高的问题。其次，在舵机控制策略方面，将弯道预判、模糊控制算法进行结合，增强了控制效果。再次，设计了独立的控制电路板，利用高精度测速传感器——编码器，将 PID 控制和模糊控制相结合实现了速度闭环控制。最后，充分利用单片机现有模块进行编程，同时拨码开关、状态指示灯、键盘、数码管等方便了算法调试。

智能车的控制系统需要硬件、软件以及机械等方面相互协调，所以在系统设计过程中需要不断的调整与完善。

关键词：智能汽车，MC9S12XS128，PID 控制，CCD 传感器，直流电机

ABSTRACT

Smart car using environmental sensing, decision-making, computer graphics technology, electronics and control technology has the abilities of premise of achieving, automatic driving and vehicle navigation support.

In this paper, we design a small intelligent car which can travel independently on the given runway, in the background of the intelligent car competition. This article firstly introduced the idea of system design, and then separately set out the design and implementation of the hardware and software, as well as the mechanical adjustment and system debugging, and finally summed up the problems encountered. Freescale 16 bit MCU served as the intelligent unit in this control system. Steering system works according to camera sensor and vehicle speed controlling relied on speed sensor .In order to improve the performance of the car, we optimize the system using some creative methods. Firstly, MC9S12XS128 used as a new single-chip controller , using the program which is compatible for digital and analog camera; the use of two parallel BTS7960 to drive the motor, a very good solution to the current driver IC drives a small, high temperature. Secondly, the steering gear controller combine curve pre-detecting and the PID algorithm. Thirdly, we design a S12XS based controlling board independently, using high-precision Rotary Encoder, and using Bang-Bang and PID control to achieve a combination of closed-loop control of speed. Lastly, we make the full use of the MCU in the process of the software design. With the help of the wireless serial port, the input switch and the denoting LED, the program debugging is easier.

Intelligent vehicle control systems require hardware, software and mechanical coordination, etc. So the design process in the system requires constant adjustment and improvement.

KEY WORDS: SMART CAR，MC9S12XS128，PID CONTROL，CCD SENSOR，DC-MOTOR

目　录

第一章　引言

1.1 智能车大赛背景以及前景

1.1.1 智能汽车的研究背景

智能汽车的研究始于 1953 年，美国 Barrett Electric 公司制造了世界上第 1 台采用埋线电磁感应方式跟踪路径的自动导向车，也被称作“无人驾驶牵引车”（Automated Guided Vehicle, AGV）。随着生产技术的发展和自动化程度的提高，传统制造业的生产方式发生了深刻的变化。为节约成本、缩短生产周期，柔性生产系统和工厂自动化等先进的生产方式逐渐发展起来。其中，AGV 则成为物流系统环节搬运设备的代表。

1.1.2 智能汽车的应用

智能车有着极为广泛的应用前景。结合传感器技术和自动驾驶技术可以实现汽车的自适应巡航并把车开得又快又稳、安全可靠；汽车夜间行驶时，如果装上红外摄像头，就能实现夜晚汽车的安全辅助驾驶；它也可以工作在仓库、码头、工厂或危险、有毒、有害的工作环境里，此外它还能担当起无人值守的巡逻监视、物料的运输、消防灭火等任务。在普通家庭轿车消费中，智能车的研发也是很有价值的，比如雾天能见度差，人工驾驶经常发生碰撞，如果用上这种设备，激光雷达会自动探测前方的障碍物，电脑会控制车辆自动停下来，撞车就不会发生了。在我国，智能车的研究也开展地如火如荼，并取得了丰硕成果。如清华大学计算机系智能技术与系统国家重点实验室在中国科学院院士张钹主持下研制的新一代智能移动机器人：THMR—V（Tsinghua Mobile Robot V）清华 V 型智能车，兼有面向高速公路和一般道路的功能。在国外智能汽车也受大很大的关注，从 2004 年起，美国国防部高级研究项目局(DARPA) 开始举办机器车挑战大赛(Grand Challenge) 。该大赛对促进智能车辆技术交流与创新起到很大激励作用。

1.2 智能车简介

为加强大学生实践，创新能力和团队精神的培养，促进高等教育的改革，受教育部高等教育司委托，高等学校自动化专业教学指导委员会负责主办全国大学生智能车竞赛。该项比赛已列入教育部主办的全国五大竞赛之一。全国大学生智能汽车竞赛原则上由全国有自动化专业的高等学校（包括港、澳地区的高校）参赛。竞赛首先在各个分赛区进行报名、预赛，各分赛区的优胜队将参加全国总决赛。每届比赛根据参赛

队伍和队员情况，分别设立光电组、摄像头组、创意组等多个赛题组别。每个学校可以根据竞赛规则选报不同组别的参赛队伍。第七届“ 飞思卡尔” 杯全国大学生智能车大赛将于 2012 年 8 月在南京师范大学举行。第七届总决赛多达 300 多支队伍参赛。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。

与前几届比赛相比，这次比赛改动比较大，摄像头组主要表现在寻迹方式由以前的跑道中间单线寻迹变为现在的跑道两边双线寻迹。在本次比赛中，参赛选手须使用大赛组委会统一提供的竞赛车模，其中摄像头组采用 A 车模，光电组采用 B 车模，电磁组采用 C 车模。采用飞思卡尔系列微控制器作为核心控制单元，自主构思控制方案及系统计，包括传感器信号采集处理、控制算法及执行、动力电机驱动、转向舵机控制等，最终实现一套能够自主识别路线，并且可以实时输出车体状态的智能车控制硬件系统。各参赛队完成智能车工程制作及调试后，于指定日期与地点参加比赛。目前，此项赛事已经成为各高校展示科研成果和学生实践能力的重要途径，同时也为社会选拔优秀的创新人才提供了重要平台。

1.3 文章的整体布局

技术报告是以智能汽车的设计为主线，具体章节如下：

第一章引言，讲述了智能车竞赛的研究背景，基本介绍，比赛规则和本报告的章节描述和文献综述。第二章智能车系统的总体设计。第三章机械结构的设计。第四章硬件的设计以及实现。第五章图像的处理。第六章软件的设计以及实现。第七章开发与调试,介绍了软件开发的环境，以及对各部分的调试方法。我们软件开发环境为 Metrowerks 公司开发的软件集成开发环境 Codewarrior，因此，我们仔细研究了 Codewarrior 使用指南。第八章总结与展望。

第二章　智能车系统总体设计

本车模的设计是以 MC9S12XS128 为核心，设计寻线方案，在稳定较高的情况下尽可能的提高车速，系统主要包括以下模块：S12 单片机，驱动电机，舵机模块，转速反馈模块和 CCD 视频采集模块。整体结构如图 2.1 所示。

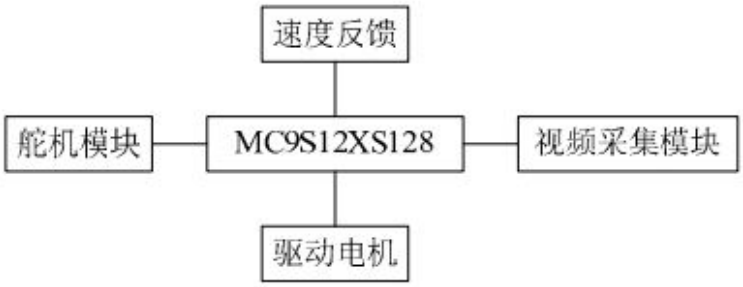


图 2.1　智能车系统功能模块图

2.1 车模的总体目标

由去年比赛的经验看，车模的可靠性是第一位的，不然车模空有很高的速度却无法跑完全程也无济于事。其次，车模在稳定工作的前提下要能达到较高的速度，并且拥有出色的转向性能。车模的可靠性包括了机械可靠性、硬件可靠性以及软件上的可靠性，其中机械的可靠性主要由严谨的机械装配和合理的机械调校来保证，而硬件和软件的可靠性主要是要保证摄像头的成像质量以及图像处理的稳定性。为了确保硬件和软件的可靠性，我们需要一个足够强大的辅助调试平台，获取足够多的测试数据，来确保其可靠。要使车模的速度提高并且有较好的转向能力，最直接的方法就是提高车模的机械性能。因此，我们需要将电路做得足够集成，从而减小车模的重量。另外，车模的重心越低对车模的性能越有利，因此摄像头不宜装在较高的位置。此外，驱动电路也是影响车模速度的关键，我们需要低内阻、大功率的驱动电路使车模获得较高的直线速度及加速度。

2.2 系统分析

本车模总的工作模式是：CCD 摄像头采集赛道信息，经过 LM1881 视频分离信号输入到 S12 单片机的 AD 转换进行转换，得到相应的图像信息。对转换所得的图像信息进一步处理，提取黑线中心等有效信息，并对舵机和电机进行控制；在舵机控制上采用的是 PD 控制；通过测速传感器反馈回来的脉冲波进行计数并计算速度；电机采用的是 PID 控制以及模糊控制。综合以上的控制方案，机械和硬件上的改善和提

高对此车模进行有效地控制，从而达在稳定性较高，转向的灵活性较好的基础上尽可能提高车的速度。系统结构图如图 2.2。

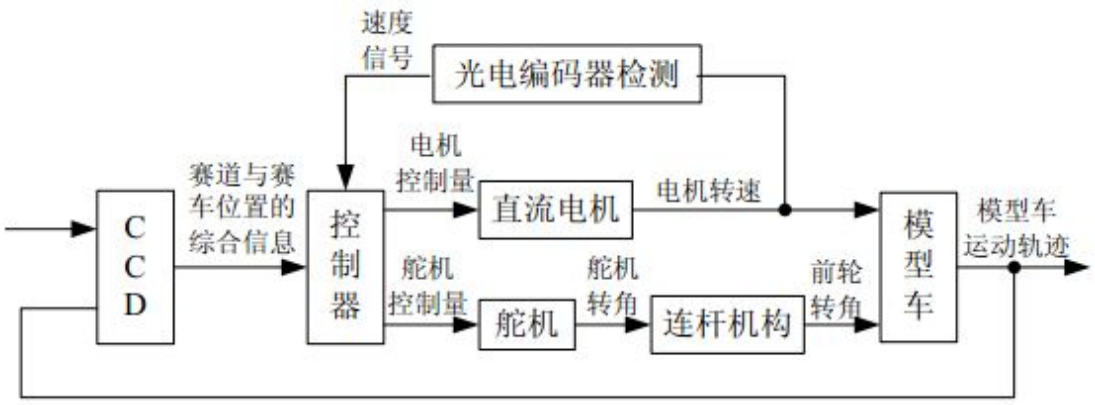


图 2.2 总体流程图

2.3 路径检测模块

路径识别模块是智能车系统的关键模块之一，路径识别方案的好坏，直接关系到最终性能的优劣。在以往比赛中光电传感器寻迹方案应用最多，单独采用 CCD 摄像头寻迹方案或者 CCD 摄像头寻迹与光电传感器寻迹结合在一起的方案也都有应用。

光电传感器寻迹方案，即路径识别电路由一系列发光二极管、接收二极管以及激光管等组成，由于赛道中存在轨迹指示黑线，落在黑线区域内的光电二极管接收到的反射光线强度与白色的赛道不同，由此判断行车的方向。光电传感器寻迹方案的优点是电路简单、信号处理速度快。但是会存在以下几个缺点：

1) 赛道空间分辨率低：一方面受到大赛规则关于传感器个数的限制；另一方面，过多的光电传感器在固定安装、占用 CPU 输入端口资源等方面也有限制；

2) 识别道路信息少；

3) 由于固定位置的限制，光电管只能安装在车模前面不远的位置，观测信息前瞻性差；

4) 容易受到环境光线的干扰；

电磁组的优点在于抗干扰能力比较强，相比于摄像头组与光电组，电磁组不受光线的影响。但是电磁组存在一个很难克服的缺点就是难以增加前瞻，在十字线出还常常识别为弯道。识别道路信息少，一般只能检测路径中心位置，赛道空间分辨率低：一方面受到大赛规则关于传感器个数的限制；另一方面，过多的磁场传感器在固定安装、占用 CPU 输入端口资源等方面也有限制。

以上光电组以及电磁组存在的问题会影响车控制精度以及运行速度。通过一定的硬软件设计，虽然可以部分解决上述的问题，但相比之下，使用面阵 CCD 器件则可以更有效的解决这些问题。CCD 在比赛规则中算作一个传感器。普通 CCD 传感器图

像分辨率都在 300 线之上，远大于光电管阵列。通过镜头，可以将车前方很远的道路图像映射到 CCD 中，从而得到车前方很大范围内的道路信息。对图像中的道路参数进行检测，不仅可以识别道路的中心位置，同时还可以获得道路的方向、曲率等信息。利用 CCD，通过图像信息处理的方式得到道路信息，可以有效进行车运动控制，提高路径跟踪精度和车运行速度。因此最后我们选择了 CCD 检测路径。

2.4 硬件系统的设计

系统硬件设计可以说是整个智能车设计的基础和重中之重。正确的硬件设计方向与思路，是系统稳定可靠的基础，功能强大的硬件系统，更为软件系统的发挥提供了保证。 同时，硬件设计的好坏会影响车模的稳定性及其转向性能，本车模硬件设计如图 2.3 所示。

2.5 软件设计模块

硬件系统以及机械都是在底层的，是整个系统的基础，而软件系统则是根据意见以及机械制定的。此模块对于整个系统来说是至关重要的，高效稳定的软件程序是车模平稳快速寻线的基础。本车软件设计的基本流程如图 2.4 所示。

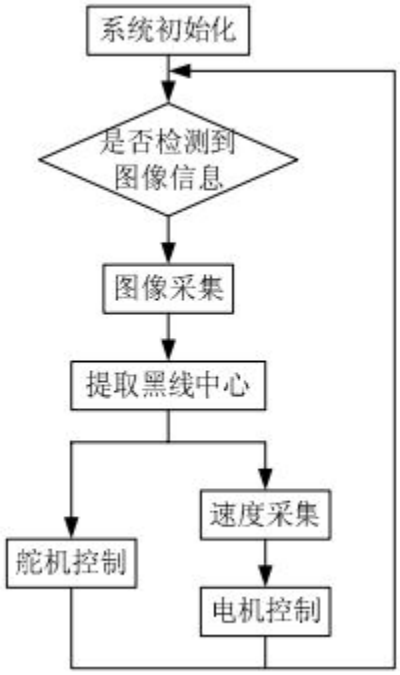


图 2.4　软件设计模块

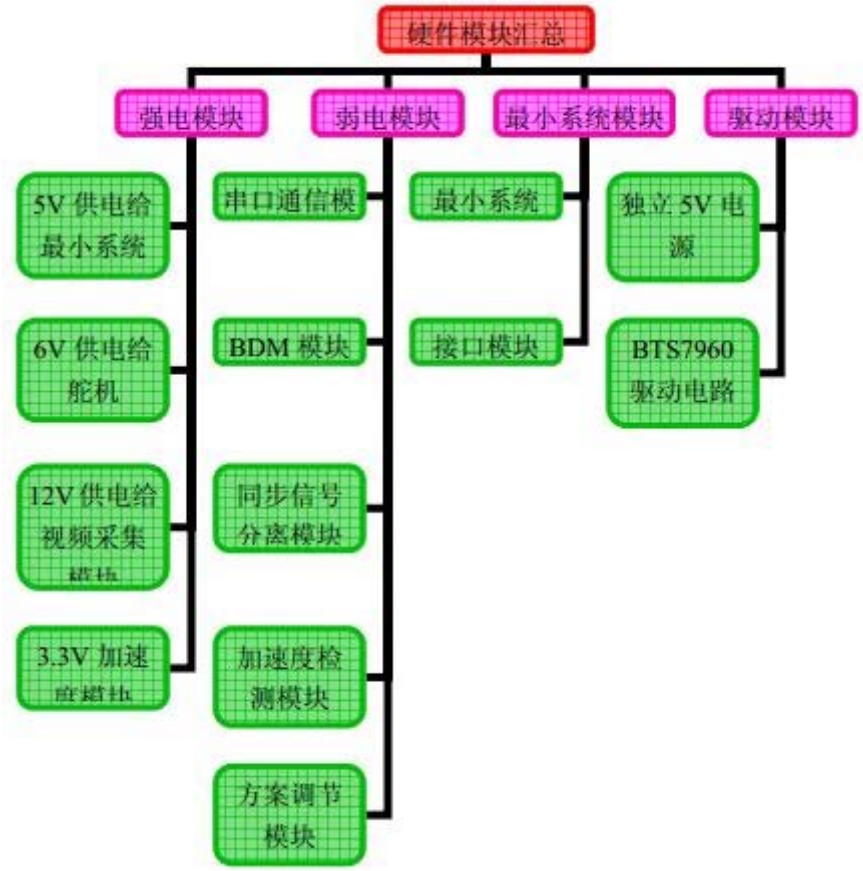

硬件模块汇总
强电模块
弱电模块
最小系统模块
驱动模块
5V 供电给最小系统
6V 供电给舵机
12V 供电给视频采集模块
3.3V 加速度模块
串口通信模
BDM 模块
同步信号分离模块
加速度检测模块
方案调节模块
最小系统
接口模块
独立 5V 电源
BTS7960 驱动电路

图 2.3 硬件模块汇总

第三章　机械系统设计

车模的机械部分是影响其行驶性能最直接的部分，其重要性不言而喻。一个不良的机械系统会增加控制的难度，会为车模的速度提升带来障碍。因此，车模的机械性能应该是优先考虑的问题。

3.1 舵机的安装

按车模说明书所说的舵机安装方法，是将舵机放在车体的右边，但是考虑到舵机的响应速度和舵机臂长，我们最后决定把舵机放到车头前，这样不但加快了舵机的响应速度，还加大了舵机打舵的力量。具体如图所示。

图 3.1　修改舵机实物图

3.2 传感器的安装

测速传感器由于本身重量很轻，对车模性能影响很小，这里就不详细介绍其安装方法了。这里重点介绍摄像头的安装。摄像头传感器重量较重，即使除去外壳用裸板，其镜头的重量对于车架来说也是不小的负担，因此摄像头的安装位置对车模的机械性能会有不小的影响。同时，摄像头安装的位置与车模的前瞻量以及视野宽度也有直接关系，所以摄像头安装的位置应同时考虑到机械性能的需要和图像的要求。从车模的性能上考虑，车模的重心越低越好，所以在图像能够接受的前提下，应尽可能降低摄像头的高度。而车体重心前后的位置保持在车体中间偏后的位置较好，因为重心太靠前车模容易甩尾，太靠后车模容易转向不足，由于车模是后轮驱动，后轮需要更大的压力来确保其动力输出，所以重心在车身中间偏后一点的位置是最理想的。从图像的角度考虑，摄像头架得低依然可以获得较远的前瞻，但视野宽度会受到限制，因

此可以通过将摄像头加在车模后方来增加其视野宽度。综合考虑以上几点，我们最终选择将摄像头架在车身中间偏后的位置，确保摄像头支架的强度和低重量。摄像头高度为 26cm。摄像头的安装图如下。

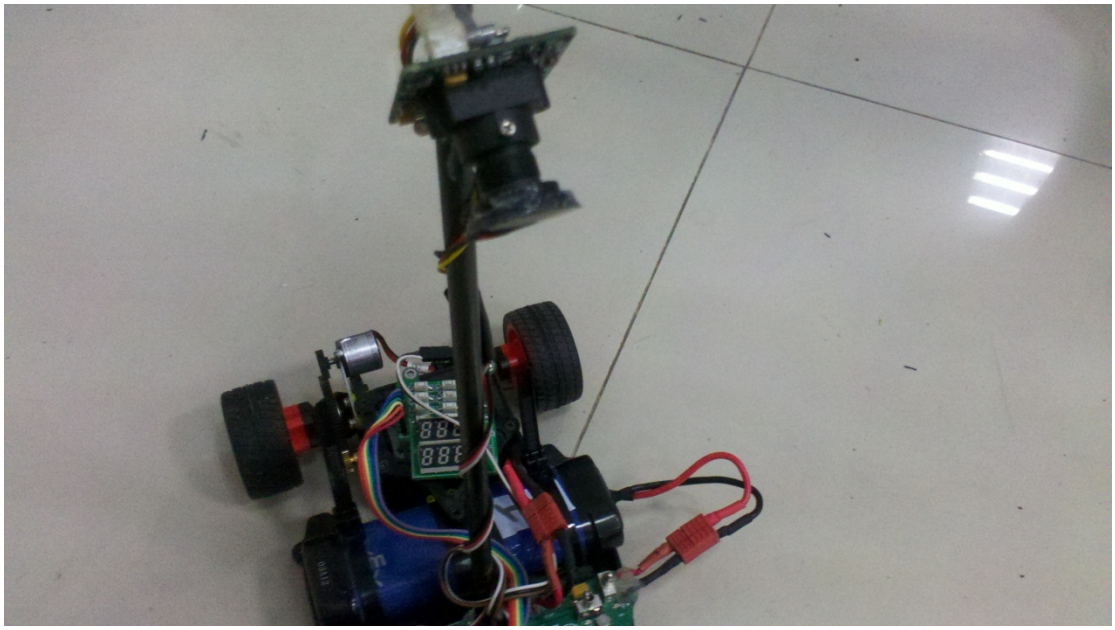


图 3.2 摄像头安装实物图

3.3 车模的机械调校

尽管比赛规则对车架做了很大的限制，但车模的机械调校仍然会起着举足轻重的作用。下面介绍几个比较关键的调校参数。

3.3.1 底盘高度

一般来说底盘越低，车模重心越低，重心移动速度越快，所以底盘越低，车模行驶越稳定，反应速度越快。考虑到赛道 KT 板厚度一般为 5mm，底盘若低于这个高度，车模在冲出赛道后可能会撞伤赛道，因此我们将车体的底盘高度设为前 5mm、后 6mm，同时将底盘前端加入防护装置，从而确保赛车的安全。

3.3.2 前悬挂角度

由于国产车模的质量不好，虚位较大，因此调整前悬挂的角度时，调节量需要大一些，以抵消虚位。在所有前悬挂的角度中，普遍认为 caster 角对转向性能影响最大。可能由于摄像头架得比较靠后的关系，车体的转向稍显不足。所以我们采用了比较极端的调校——4°caster，使车头侧倾量增加，从而增加车体的转向能力，其缺点是降低了转向的反应速度，使连续弯中的性能下降。

3.3.3 前轮倾角的调整

我们在调试中发现：由于前轮轴和车轮之间的间隙较大，对车高速转向时的中心

影响较大，会引起高速转向下车的转向不足。而且这里又是规则中严禁改动的部分，所以为了尽可能降低转向舵机负载，我们对前轮的安装角度，即前轮定位进行了调整。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。在实际调试中，我们发现适当增大内倾角的确可以增大转弯时车轮和地面的接触面积，从而增大车了地面的摩擦程度，使车转向更灵活，减小因摩擦不够而引起的转向不足的情况。

3.3.4 齿轮传动机构及后轮差速的调整

车模后轮采用 RS-380-ST/3545 电机驱动，由竞赛主办方提供。电机轴与后轮轴之间的传动比为 9:38（电机轴齿轮齿数为 18，后轮轴传动轮齿数为 76）。齿轮传动机构对车模的驱动能力有很大的影响。齿轮传动部分安装不恰当，会增大电机驱动后轮的负载；齿轮配合间隙过松则容易打坏齿轮过紧则会增加传动阻力。所以我们在电机安装过程中尽量使得传动齿轮轴保持平行，传动部分轻松、流畅，不存在卡壳或迟滞现象。差速机构的作用是在车模转弯的时候，降低后轮与地面之间的滑动；并且还可以保证在轮胎抱死的情况下不会损害到电机。差速器的调整中要注意滚珠轮盘间的间隙，过松过紧都会使差速器性能降低，转弯时阻力小的车轮会打滑，从而影响车模的过弯性能。好的差速结构，在电机不转的情况下，右轮向前转过的角度与左轮向后转过的角度之间误差很小，不会有迟滞或者过转动情况发生。

第四章　硬件系统设计

4.1 硬件设计方案

从最初进行硬件电路设计时我们就既定了系统的设计目标：可靠、高效、简洁，在整个系统设计过程中严格按照规范进行。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。我们主要是从以下两个方面实现的：

1. 将不同的功能模块放在不同的电路板上，在出现问题的时候能够逐一的进行检查和修正，大大的缩短了检查问题和解决问题的时间。

2. 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 100A 以上，大大提高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。我们在对电路进行详细、彻底的分析后，对电路进行了大量简化，并合理设计元件排列和电路走线，使本系统硬件电路部分轻量化指标都达到了设计要求。

4.2 电路设计方案实现

整个智能车控制系统是由四部分组成的：升压稳压电源板、S12X 为核心的最小系统板、主板、电机驱动电路板。最小系统板可以插在主板上组成信号采集、处理和电机控制单元。为了减小电机驱动电路带来的电磁干扰，我们把控制单元部分和电机驱动部分分开来，做成了两块电路板。主板上装有组成本系统的主要电路，它包括如下部件：视频同步分离电路、加速度传感器模块、摄像头接口、舵机接口、电机驱动器接口、编码器接口、电源接口、单片机最小系统板插座、跳线、指示灯、按键、开关等。

4.2.1 单片机最小系统

单片机最小系统部分使用 MC9S12XS128 单片机 112 引脚封装，为减少电路板空间，仅将本系统所用到的引脚引出，包括 PWM 接口，计数器接口，外部中断接口，若干普通 IO 接口。其他部分还包括电源滤波电路、时钟电路、复位电路、串行通讯接口、BDM 接口和 SPI 接口。为简化电路，我们取消了复位按键和串行通讯接口电路中的 TTL 电平与 RS-232 电平转换电路。单片机最小系统板电路原理图如图 4.1。

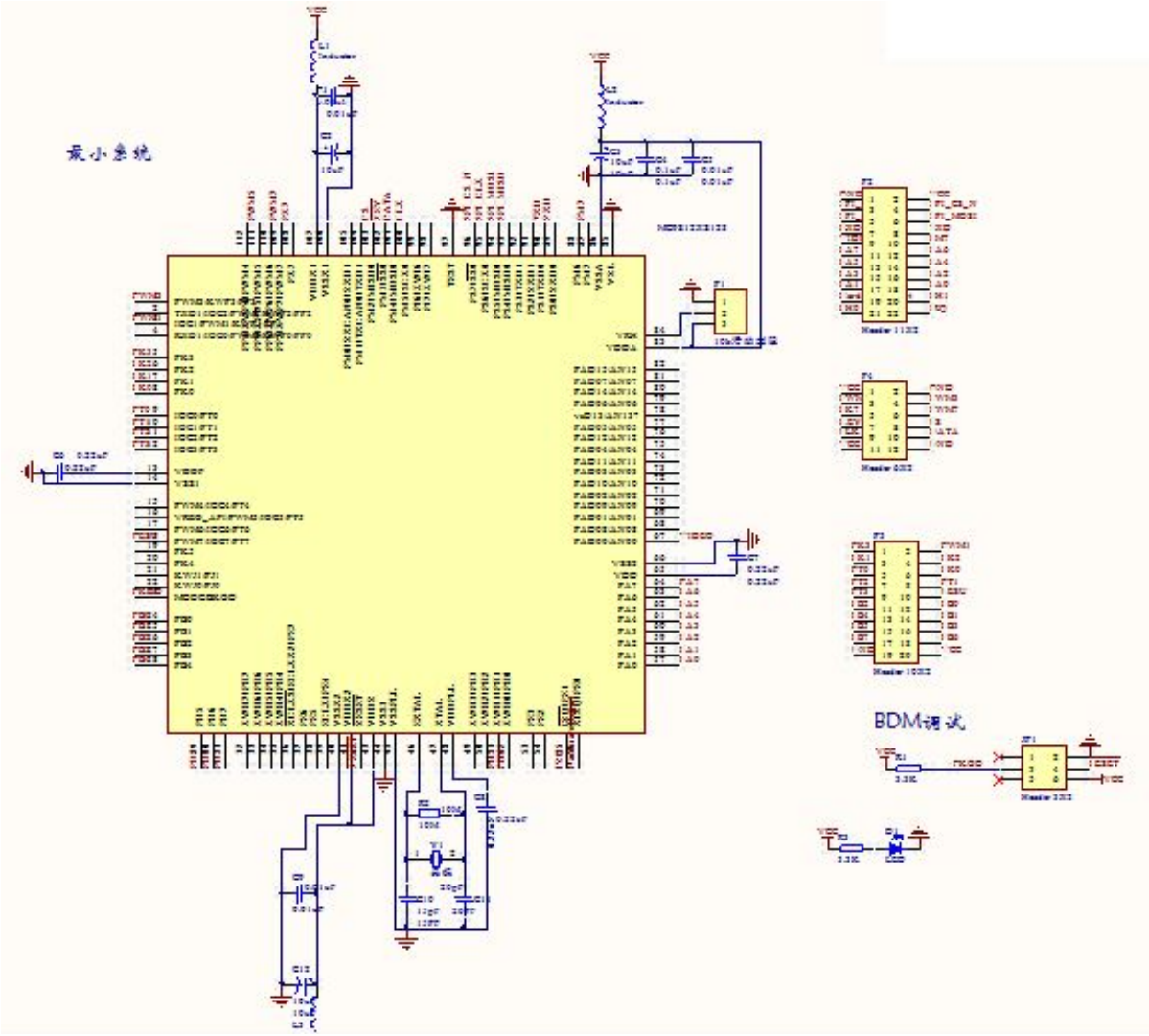


图 4.1　单片机最小系统板电路原理图

单片机引脚规划如下：

PAD0：视频模拟信号输入

PORTA0-5：方案调节信号输入

PORTB0-7：LED　测试灯

PT2：奇偶场信号

IRQ：行同步脉冲输入信号。

PT7：光电编码器脉冲输入信号。

PWM7：舵机角度控制信号输出。

PWM0-3：电机速度控制信号输出。

PS0-1：SPI 通讯信号

4.2.2 电源稳压电路

本系统中电源稳压电路有四路，一路为+5V 稳压电路，一路为+12V　稳压电路，一路为 6V 稳压，一路为 3.3V 稳压。由于整个系统中+5V 电路功耗较小，为了降低

电源纹波，我们考虑使用线性稳压电路。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，常用的低压降串联稳压芯片主要有 2940、1117 等等。2940 虽然压降比 1117 更低,但是纹波电压较大。相比之下，1117 的性能更好一些,具有输出电压恒定，压降较低的优点，但是其线性调整工作方式在工作中会造成较大的热损失，导致电源利用率不高，工作效率低下。

TPS7350 是微功耗低压差线性电源芯片,具有完善的保护电路,包括过流,过压,电压反接保护。使用这个芯片只需要极少的外围元件就能构成高效稳压电路。与前两种稳压器件相比，TPS7350 具有更低的工作压降和更小的静态工作电流，可以使电池获得相对更长的使用时间。由于热损失小，因此无需专门考虑散热问题。

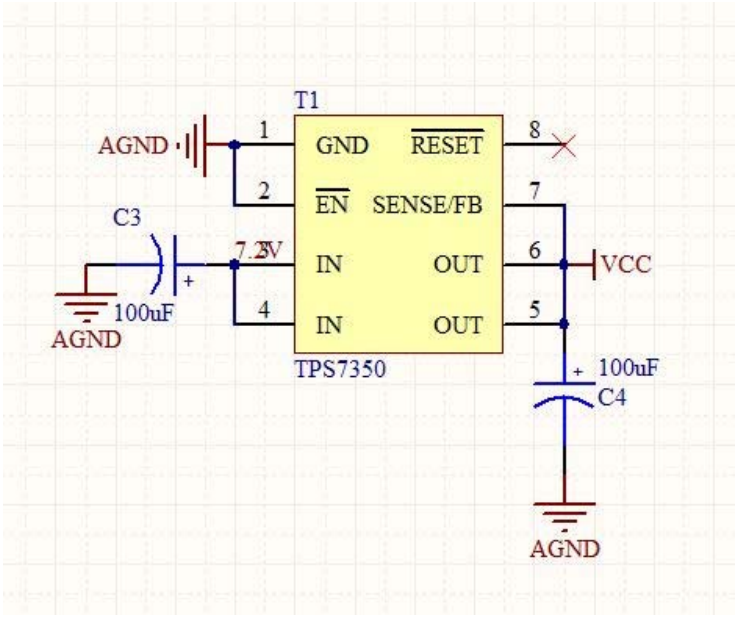


图 4.2 TPS7350 电源稳压电路

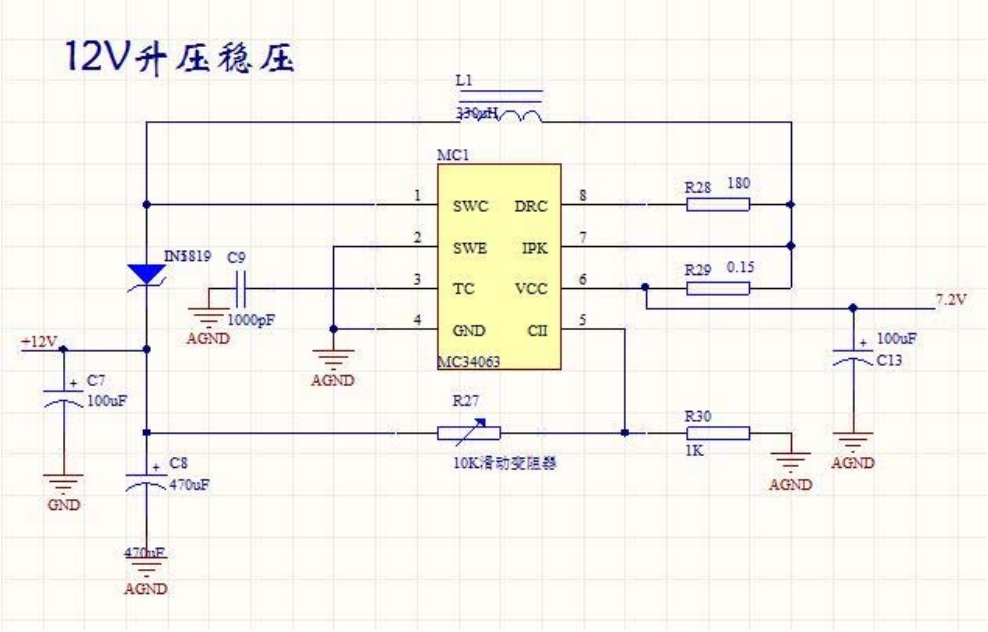


图 4.3 MC34063 +12V 升压电路

MC34063 是一单片双极型线性集成电路,专用于直流-直流变换器控制部分。片内包含有温度补偿带隙基准源、一个占空比周期控制振荡器、驱动器和大电流输出开关,能输出 1.5A 的开关电流.它能使用最少的外接元件构成开关式升压变换器、降压式变换器和电源反向器。

LM1117-3.3V 是 3.3V 稳压芯片，LM1117 有两个型号，分别是 5V 稳压和 3.3V 稳压，3.3V 稳压芯片主要是用于为加速度传感器 MMA7260 供电。

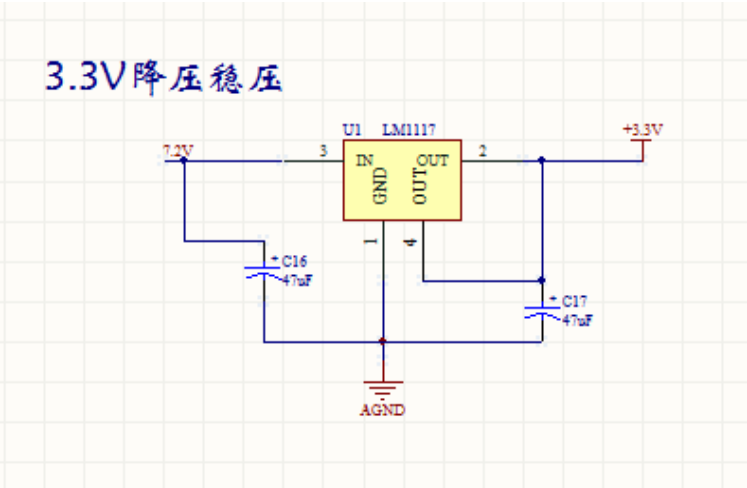


图 4.4　LM1117 +3.3V 稳压电路

4.2.3 视频同步分离电路

我们的智能模型车自动控制系统中使用黑白全电视信号格式 CCD 摄像头采集赛道信息。摄像头视频信号中除了包含图像信号之外，还包括了行同步信号、行消隐信号、场同步信号、场消隐信号以及槽脉冲信号、前均衡脉冲、后均衡脉冲等。因此，若要对视频信号进行采集，就必须通过视频同步分离电路准确地把握各种信号间的逻辑关系。我们使用了 LM1881 芯片对黑白全电视信号进行视频同步分离，得到行同步、场同步信号,具体原理不再赘述。视频同步分离电路原理图 4.5 所示。

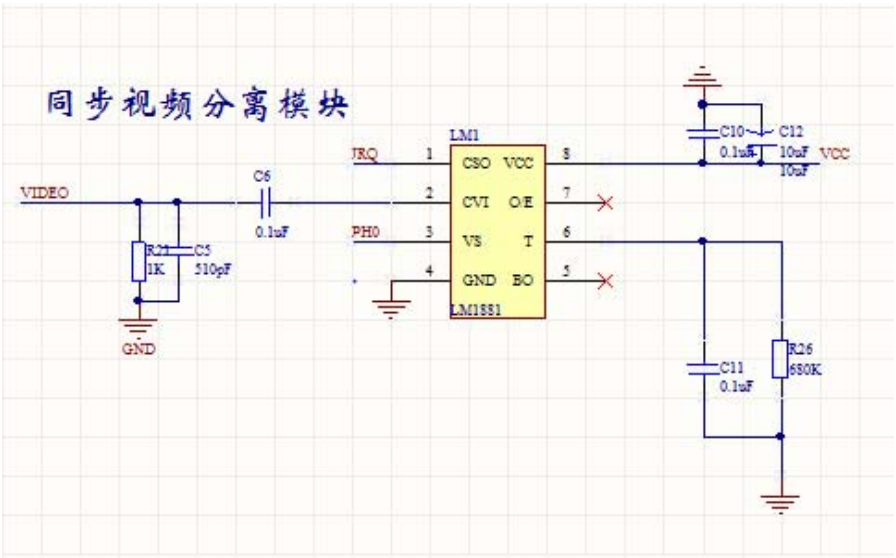


图 4.5　视频同步分离电路

4.2.4 电机驱动电路

电机驱动板为一个由分立元件制作的直流电动机可逆双极型桥式驱动器，其功率元件由四支 N 沟道功率 MOSFET 管组成，额定工作电流可以轻易达到 100A 以上，大大提高了电动机的工作转矩和转速。该驱动器主要由以下部分组成： PWM 信号输入接口、逻辑换向电路、死区控制电路、电源电路、上桥臂功率 MOSFET 管栅极驱动电压泵升电路、功率 MOSFET 管栅极驱动电路、桥式功率驱动电路、缓冲保护电路等。

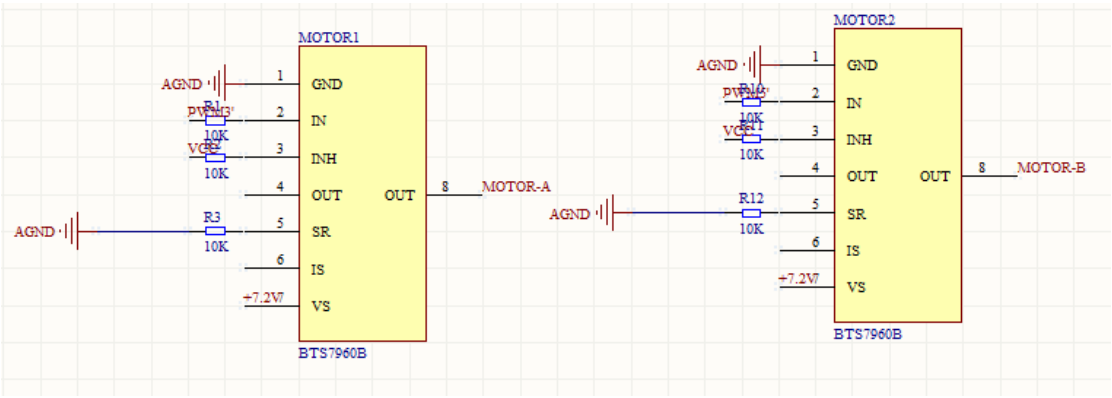


图 4.6 电机驱动电路

4.2.5 PCB 电路板

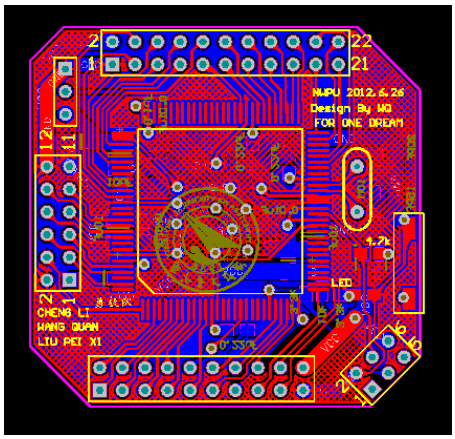


图 4.7 最小系统 PCB 板

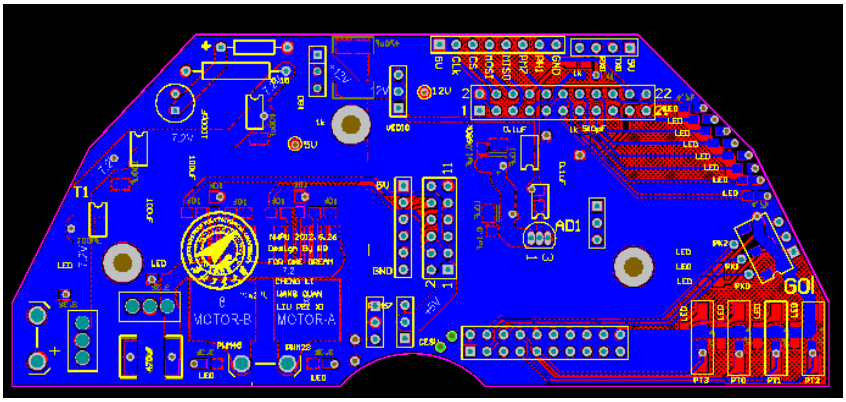


图 4.8 主板

4.3 传感器的选择

4.3.1 摄像头

目前市面上常见的摄像头主要有 CMOS 和 CCD 两种，CCD 摄像头具有对比度高、动态特性好的优点，但需要工作在 12V 电压下，对于整个系统来说过于耗电，且图像稳定性不高，由于在使用过程中温度的升高，所采的 AD 值也会随之而变化；CCD 摄像头体积小，耗电量小，虽然 CCD 摄像头我们也采用了 12V 的供电但是由于工作电流非常的小，功率很小不会出现明显的发热，图像稳定性较高，黑白对比度非常的明显。因此，经过实验论证之后我们决定采用 CCD 摄像头。最终采用的 SS2000A-2/B-2 摄像头具体参数如下：摄像头参数：320 线；输出制式：PAL 制式标准视频信号；供电电压/ 消耗功率：12V 100mA 。

4.3.2 测速编码器

根据以往的经验我们使用的都是欧姆龙的测速编码器，但是考虑到欧姆龙的测速编码器体积比较大，安装起来比较麻烦。而且相比于我们所用到的精度，欧姆龙的精度比较的高为 320 线，用 100 线的测速编码器完全可以达到要求，所以我们选取了小型达到测速编码器，码盘内部使用发光二极管等电子元件组成的检测装置检测输出脉冲信号，通过计算每秒光电编码器输出脉冲的个数就能反映当前电动机的转速。最后还是根据不同车的需要进行了选择。测速编码器的实物图如下。

图 4.9　测速编码器实物图

第五章　图像处理

5.1 提取黑线

5.1.1 视频采集原理

视频信号采集模块是由摄像头，LM1881 视频信号分离芯，MC9S12XS128 单片机的 AD 转换模块等组成。图像采集的原理是根据视频同步信号对单个奇场或偶场图像信号进行隔行、隔点采集，一方面为了保证时序上的准确性，必须准确的判断同步信号和严格的定时，另一方面定时的时间相对于 CPU 周期是比较长的，在这些时间内可以让 CPU 去做其它工作，故在此部分中的程序多采用了中断。采集一场图像的流程如下：

第一步：LM1881 分离出的奇场或偶场同步信号在 MC9S12XS128 的 PT7 口产生中断，通过 ECT 捕捉中断，即等待场信号，此时还有 22.5 行（约 1408μs ，视 LM1881 的信号延时而定）将开始第一行图像。为确保采集到的信号是准确无误的，这时利用定时器延时，等到延时结束之后，开启行同步中断。

第二步：大约 32μs （0.5 行）后，行中断到。由于一场中有 280 多个行，但 AT D 只采集其中的 80 行，为此设置了一个行计数器，每次行中断都将该计数器加一，判断行数是否已经满足，若是满足则采集完成，否则，根据 PA L 的规范，在数据行中，行同步信号后 6μs（上升沿，如果选下降沿有效，应该是 10.3μs ）才会有真正的图像数据出现，并持续 52μs 到该行结束。因此要利用定时器延时，跳过行同步信号和消隐信号。

第三步：对一行视频信号进行连续采集；延时，跳过 6.5 行视频信号即延时 416us，跳回到等待行同步信号，直至完成。

5.1.2 图像的处理

本届大赛跑道与以往相比有很大的改动，有单线寻迹变成了双线寻迹，但是图像的处理还是一样的原理。如图 5.1

图像检测范围与图像复杂程度是一组矛盾。若图像检测范围大，在转弯等一些情况下在图像的边界或角落里就会出现干扰，不利于在软件上对图像的处理。若令图像检测的范围减小以使在所有情况下图像中都不会出现干扰，结果会出现如下情况：有些转弯采集不到黑线。

检测到的直线赛道与弯道的差别很小，不利于算法上区分。

因此在系统设计过程中，先把 CCD 信号接入电视，在多个不同位置多次试验得出一个 CCD 摄像头安装最佳位置，使得检测范围足够大且干扰尽量小。

在有干扰的情况下，经过测试发现，一般在图像中最后一行像素点数据即离小车最近的一行出现干扰的情况极小，故可以在此行内搜索第一个黑点作为黑线左边界位置，搜索结束，以这个位置为基准左右各偏移一定距离作为下一行搜索的范围，下一行搜到边界后，再依次类推计算出以后各行的搜索范围。

在有灯光反射干扰的情况下，有些行的图像中无黑线，若在此行搜索范围内搜索不到黑线，则记录一个错误位置标志，下一行搜索范围在此行基础上再增加一些。当一幅图像中搜索的结果中错误标志过多时，就认为此幅图像出错，控制方案按照上一幅图像的处理结果决定。

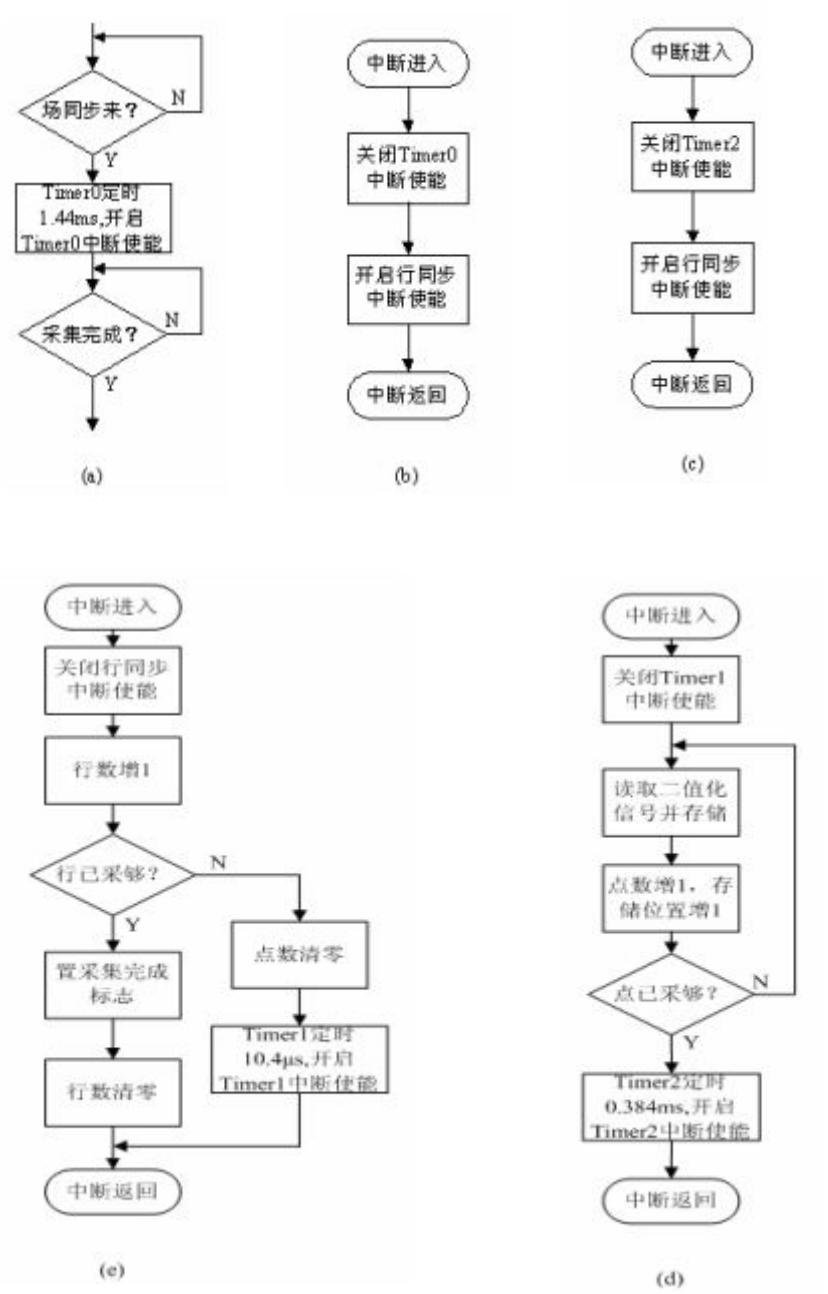


图 5.1 （a ）主程序图像采集部分（b）Timer0 中断程序（c ）Timer2 中断程序（d ）Timer1 中断程序 （e ）行同步

用上面提出的方法来处理采集进来的图像，可以得到如图 5.2 的结果。

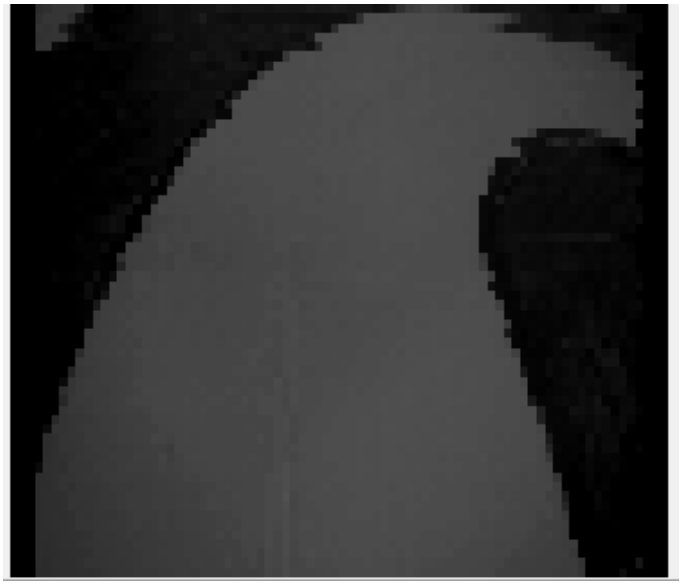


图 5.2

试验结果表明，此种方法能够有效地排除干扰，找到黑线的左边界，且效率高，处理时间短。视频去噪处理在再现真实的路径方面有着不可忽视的作用，正是由于 CCD 视觉传感器易受环境影响，才需要去噪算法进行去噪处理，保证视频数据的准确性。

在本系统中，我们采取的视频去噪策略是取距离车体最近的数据，也就是第一行的数据作为参考数据，根据数据的连续性，进行整场数据的噪声处理。本模型车检测黑线中心位置算法的大体思想如下：首先判断图像采集是否完成，完成了才开始下面的计算。从最近一行图像开始检测，设定检测的左、右边界值。

在边界值限定的范围内检测各个点是否是黑点，并记录下黑点的左、右边界位置。根据该行获得的左、右黑线边界位置来确定下一行检测的左、右边界值。如此循环直到处理完整幅图像。此外，考虑到在某些行上可能没有测到黑点，这种情况的发生有可能是图像本身就没有黑点，也有可能是错误的信息。

因此，对于没有黑点的行将置一个无效标志位，为之后能成功地对电机及舵机进行控制做准备。黑线中心的检测算法流程图如图 5.3 所示。

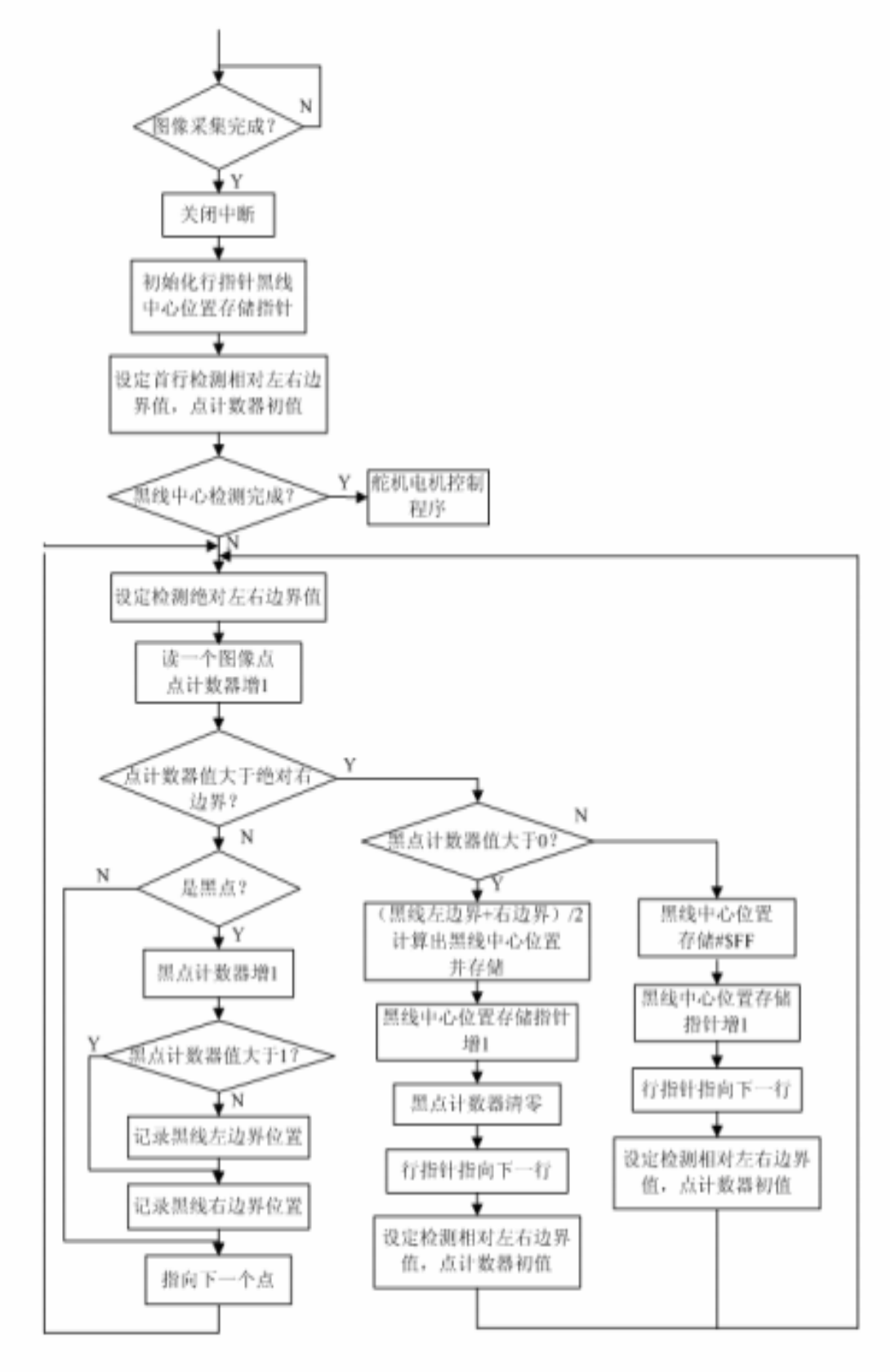
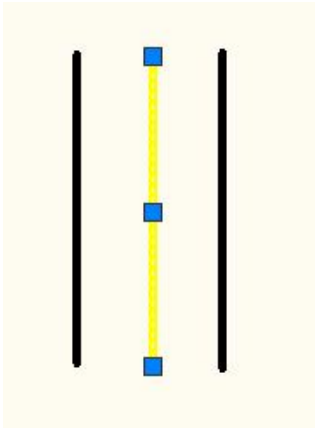

图像采集完成?
N
Y
关闭中断
初始化行指针黑线中心位置存储指针
设定首行检测相对左右边界值，点计数器初值
黑线中心检测完成?
Y
舵机电机控制程序
N
设定检测绝对左右边界值
读一个图像点
点计数器增1
点计数器值大于绝对右边界?
Y
N
黑点计数器值大于0?
N
Y
是黑点?
N
Y
黑点计数器增1
黑点计数器值大于1?
Y
N
记录黑线左边界位置
记录黑线右边界位置
指向下一个点
（黑线左边界+右边界）/2
计算出黑线中心位置
并存储
黑线中心位置存储指针增1
黑点计数器清零
行指针指向下一行
设定检测相对左右边界值，点计数器初值
黑线中心位置
存储#SFF
黑线中心位置存储指针增1
行指针指向下一行
设定检测相对左右边界值，点计数器初值

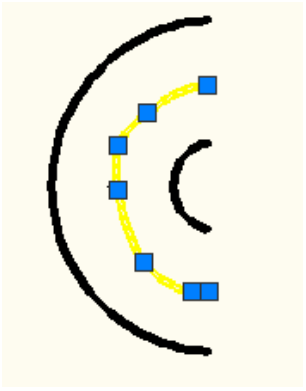
图 5.3　黑线提取流程图

5.2 均值的计算

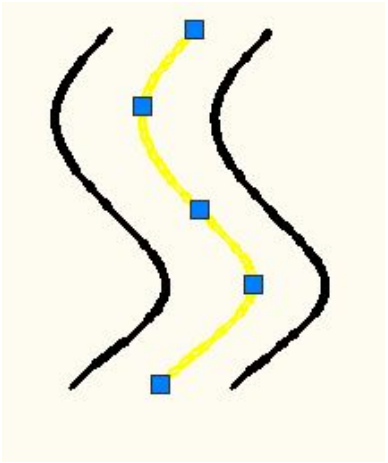
5.2.1 赛道的形状



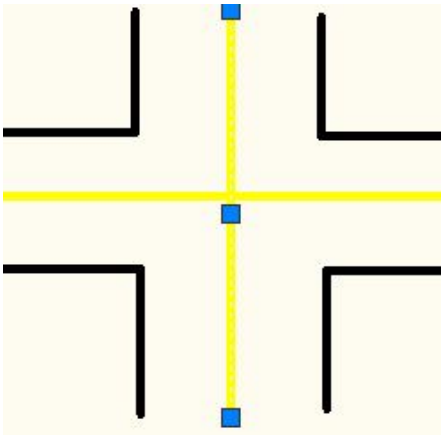
(a)直道



(b)急转弯



(c)”S”形弯



(d)十字交叉

图 5.4 赛道形状模拟图

如图 5.4 所示，比赛道路主要分为直道、急转弯、“S” 形弯、十字交叉四种形状，其中黑线为跑道两侧黑色引导线，黄色为由两条黑线虚拟出来的跑道中线。

虽然比赛中有小于 15º 的上下坡，但配合控制算法，上下坡可以近似认为是直道。为了高效的通过这些形状的道路，首先必须正确识别不同的道路形状。

5.2.2 赛道的识别

在 CCD 摄像头已经准确识别道路两侧黑线的基础上，利用所提取的信息，经过处理后可以分辨出当前路面的形状。具体方法是均匀选取若干行的虚拟黑点中心位置，计算出所选 n 个黑点的位置 x 相对于视场中心 View_Centre 的平均位置$\bar{x}$， 计算公式如下。

$$\bar{X} = \frac{\sum_{i=1}^{n} x_i}{n}$$

这样计算出来的黑点中心的平均位置可以用来分辨出不同的赛道，而且其优点是，如果区分各种图形的阈值设定得当，这种方法能够将图中所示的“S”形弯识别为直线，这正是我们想要的结果。X 与视场中心 View_Centre 的绝对值之差 Offset 表示当前赛道的弯曲程度，其值的正负表示是向左弯曲还是向右弯曲。值越大，说明拐弯半径越小，若遇急转弯，则 Turn_Flag 置位。

实际试验中发现，以上的算法对于“S”形弯、直道、弯道区分效果理想，在车体平行驶过十字交叉路段时，也可以正确区分十字交叉路，在 5.1 小节中所述的黑线提取中，提取到以下信息：

1) 虚拟黑线中心位置
2) 有效行数
3) 虚拟黑线的变化幅度

利用这些信息可以识别赛道的路劲，要想取得好的成绩，对于摄像头组来说可以提前转弯，小 S 取直等优化路劲。均值的计算需要一定的技巧，一般来说都是采用以下这个公式 10 进行计算的：

具体程序如下：

```
for(i=0;i++;i<=60) {
if(black_wire[i]!=0XFF)
{sum=sum+black_wire[i]; lcount++;}}
X_PRESENT =sum/lcount;
```

black_wire[i]存储虚拟黑线中心的数组；lcount 是有效行计数器；X_present 是整幅图像的均值；均值大小可以判断赛道路劲的变化，这是赛道的基本特征。

但是这样计算不利于判断是否在弯道内，因为用以上的方法计算出来的均值接近于视场均值，因此会出现误判，为了更好的解决这个问题，利用有效行，以及最远行与最近行的均值的等信息加以判断，实验证明这个方法比上述方法更准的判断赛道的形状。

```
X_PRESENT=X_PRESENT+(black_w ire[lcount-1]-black_wire[0])
*line_near_n/line_near_d;
```

line_near 是视场近处的行数，line_near_d 是一个比例常数。

第六章　软件系统设计

6.1 系统初始化

本系统是以 MC9S12XS128 单片机为核心，尽可能利用其各个功能模块，在编写算法之前，先初始化 MC9S12XS128 单片机。既是对 MC9S12XS128 单片机各个寄存器,，各端口进行设置，并定义自变量，分配存储空间，使之满足系统要求。详细信息请查看文献。

工作模式：通过软件与硬件的结合，选定单片机工作模式为普通弹片模式。

时钟初始化：单片机内部的总线频率为 24MHz，CPU 单元工作频率是频率的 2 倍为 48MHz。

```
REFDV=1;
SYNR=5;                    //总线时钟=16MHz*（SYNR+1 ）/
（REFDV+1)=48MHz
while(CRGFLG_LOCK==0);          //等待 VCO 达到稳定
CLKSEL=0x80;                     // 打开 PLL
DisableInterrupts;
```

存储空间分配：对内部地址资源的分配采用普通单片工作模式初始化时默认的配置，即$0000 到$0400 为寄存器地址空间，$2000 到$3FFF 为内部 RAM 地址空间，$4000 到$7FFF 为一块固定的 Flash EEPROM 地址空间，$8000 到$BFFF 为页面 Flash EEPROM 地址空间，$C000 到$FFFF 为一块固定的 Flash EEPROM 地址空间，其中$FF00 到$FFFF 为中断向量地址空间。

AD 转换初始化：采集图像时，AD 转换的初始化是必不可少的，根据 ATDClock=[BusClock*0.5]/[PRS+1]式子得到 AD 转换分得的频率。以下是其初始化程序：

```
ATD0CTL1=0x00;     //7:1-外部触发,65:00-8 位精度,4:放电,3210:ch
ATD0CTL2=0x40;     // 禁止外部触发,  中断禁止
ATD0CTL3=0x08;     // 左对齐无符号
ATD0CTL4=0x00;     //765: 采样时间为 4 个 AD 时钟周期,
ATD0CTL5=0x00;
ATD0DIEN=0x00;     // 禁止数字输入
```

复用端口设置:

a) A 端口为普通输入端口

b) T 端口为中断输入端口

c) P 端口为 PWM 信号输出端口

还有定时模块的初始化，PWM 模块，ECT 模块的初始化，详细程序请查看附录。

```
SEGMENTS /* here all RAM/ROM areas of the device are listed. Used in PLACEM
/* Register space  */
/*    IO_SEG         = PAGED                              0x0000 TO   0x07FF;

/* non-paged RAM */
      RAM            = READ_WRITE  DATA_NEAR              0x2000 TO   0x3FFF;

/* non-banked FLASH */
      ROM_4000       = READ_ONLY   DATA_NEAR IBCC_NEAR    0x4000 TO   0x7FFF;
      ROM_C000       = READ_ONLY   DATA_NEAR IBCC_NEAR    0xC000 TO   0xFEFF;
/*    VECTORS        = READ_ONLY                          0xFF00 TO   0xFFFF;
    //OSVECTORS      = READ_ONLY                          0xFF10 TO   0xFFFF;

/* paged EEPROM                                          0x0800 TO   0x0BFF;
      EEPROM_00      = READ_ONLY   DATA_FAR IBCC_FAR   0x000800 TO 0x000BFF;
      EEPROM_01      = READ_ONLY   DATA_FAR IBCC_FAR   0x010800 TO 0x010BFF;
      EEPROM_02      = READ_ONLY   DATA_FAR IBCC_FAR   0x020800 TO 0x020BFF;
      EEPROM_03      = READ_ONLY   DATA_FAR IBCC_FAR   0x030800 TO 0x030BFF;
      EEPROM_04      = READ_ONLY   DATA_FAR IBCC_FAR   0x040800 TO 0x040BFF;
      EEPROM_05      = READ_ONLY   DATA_FAR IBCC_FAR   0x050800 TO 0x050BFF;
      EEPROM_06      = READ_ONLY   DATA_FAR IBCC_FAR   0x060800 TO 0x060BFF;
      EEPROM_07      = READ_ONLY   DATA_FAR IBCC_FAR   0x070800 TO 0x070BFF;

/* paged RAM:                                            0x1000 TO   0x1FFF;
/*    RAM_FE         = READ_WRITE                       0xFE1000 TO 0xFE1FFF;
/*    RAM_FF         = READ_WRITE                       0xFF1000 TO 0xFF1FFF;

/* paged FLASH:                                          0x8000 TO   0xBFFF;
      PAGE_F8        = READ_ONLY   DATA_FAR IBCC_FAR   0xF88000 TO 0xF8BFFF;
      PAGE_F9        = READ_ONLY   DATA_FAR IBCC_FAR   0xF98000 TO 0xF9BFFF;
      PAGE_FA        = READ_ONLY   DATA_FAR IBCC_FAR   0xFA8000 TO 0xFABFFF;
      PAGE_FB        = READ_ONLY   DATA_FAR IBCC_FAR   0xFB8000 TO 0xFBBFFF;
      PAGE_FC        = READ_ONLY   DATA_FAR IBCC_FAR   0xFC8000 TO 0xFCBFFF;
/*    PAGE_FD        = READ_ONLY                       0xFD8000 TO 0xFDBFFF;
      PAGE_FE        = READ_ONLY   DATA_FAR IBCC_FAR   0xFE8000 TO 0xFEBFFF;
/*    PAGE_FF        = READ_ONLY                       0xFF8000 TO 0xFFBFFF;
END
```

图 6.1　单片机内存空间分布图

6.2 控制算法

常见的控制算法有两种模糊控制和 PID 控制。

6.2.1 模糊控制

模糊控制实质上是用计算机去执行操作人员的控制策略，因而可以避开对像复杂的数学模型，力图对人们关于某个控制问题的成功与失败的经验进行加工，总结出知识，从中提炼出控制规则，实现复杂系统的控制。其特点为：

1)　模糊工程的计算方法虽然是运用模糊集理论进行的模糊算法，但最后得到的控制规律是确定性的、定量的条件语句。

2)　不需要根据机理与分析建立被控对像的数学模型，对于某些系统，要建立数学模型是很难的，甚至是不可能的。

3)　与传统的控制方法相比，模糊控制系统依赖于行为规则库，由于是用自然语言表达的规则，更接近于人的思维方法和推理习惯，因此，便于现场操作人员的理解

和使用，便于人机对话，以得到更有效的控制规律。

4) 模糊控制与计算机密切相关。从控制角度看，它实际上是一个由很多条件语句组成的软件控制器。目前，模糊控制还是应用二值逻辑的计算机来实现，模糊规律经过运算，最后还是进行确定性的控制。模糊推理硬件的研制和模糊计算机的开发，使得计算机将像人脑那样随心所欲地处理模棱两可的信息，协助人们决策和进行信息处理。

模糊控制器是模糊控制系统的核心，是模糊控制系统控制品质的主要保证，因此，在模糊控制系统中，设计和调整模糊控制器的工作是很重要的。模糊控制是以控制人员的经验为基础实施的一种智能控制，它并不需要精确的数学模型去描述系统的动态过程，因此，它的设计方法与常规控制器的设汁方法有所不同。模糊控制器的设计，一般是先在经验的基础上确定各个相关参数及其控制规则，然后在运行中反复进行调整，以达到最佳控制效果。

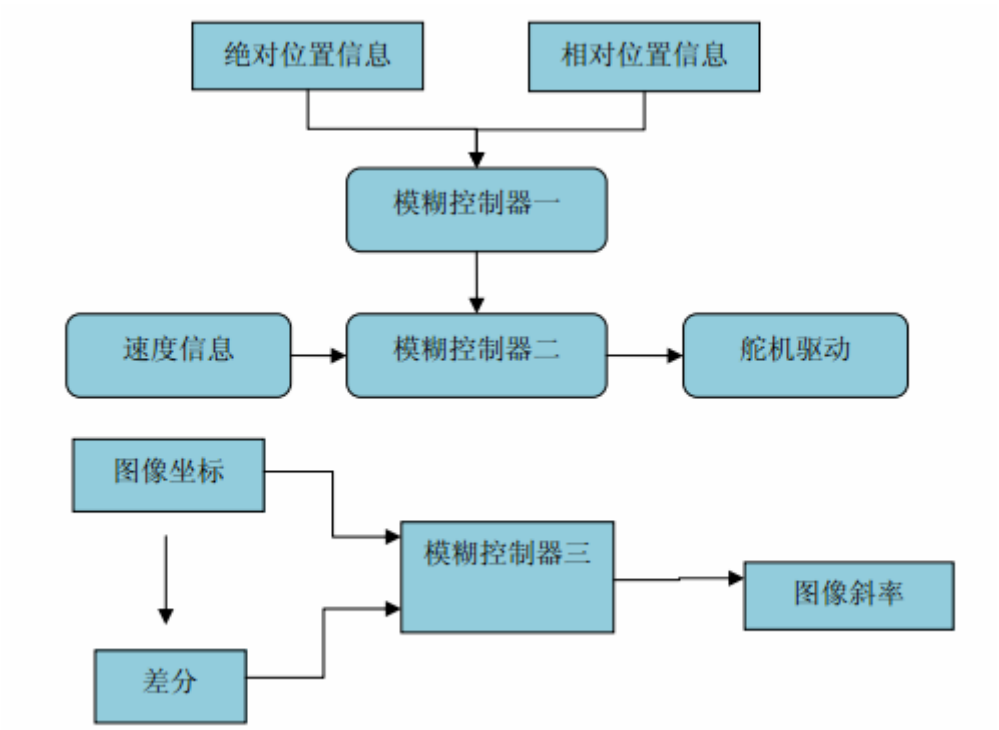


图 6.2 转向模糊控制器设计

模糊控制在横向上，即对小车的方向控制，可以采用了由采集图像上的绝对位置、相对位置和模型车速度三个变量组成的模糊控制器来控制；在纵向上，即对模型车的速度控制，可以采用了基于模型车当前速度和转角为输入的模糊控制。图中的绝对位置信息为黑线与图像中心线的距离值，相对位置信息为黑线的斜率值，速度量为速度传感器的反馈值。

6.2.2 经典 PID 控制算法

PID （Proportional Integral Differential）控制是比例、积分、微分控制的简称.在自动控制领域中，PID 控制是历史最久、生命力最强的基本控制方式。问世至今 70 多年，它以结构简单，稳定性好，工作可靠等优点而被应用在工程控制领域。

PID 控制器的参数整定是控制系统设计的核心内容。它是根据被控过程的特性确定 PID 控制器的比例系数、积分时间和微分时间的大小。PID 控制器参数整定的方法很多，概括起来有两大类：一是理论计算整定法。它主要是依据系统的数学模型，经过理论计算确定控制器参数。这种方法所得到的计算数据未必可以直接用，还必须通过工程实际进行调整和修改。二是工程整定方法，它主要依赖工程经验，直接在控制系统的试验中进行，且方法简单、易于掌握，在工程实际中被广泛采用。PID 控制器参数的工程整定方法，主要有临界比例法、反应曲线法和衰减法。三种方法各有其特点，其共同点都是通过试验，然后按照工程经验公式对控制器参数进行整定。但无论采用哪一种方法所得到的控制器参数，都需要在实际运行中进行最后调整与完善。例如：采用临界比例法。利用该方法进行 PID 控制器参数的整定步骤如下：

(1) 首先预选择一个足够短的采样周期让系统工作；

(2) 仅加入比例控制环节，直到系统对输入的阶跃响应出现临界振荡，记下这时的比例放大系数和临界振荡周期；

(3) 在一定的控制度下通过公式计算得到 PID 控制器的参数。

在实际调试中，只能先大致设定一个经验值，然后根据调节效果修改。

对于温度系统：P（% ）20--60 ，I（分）3--10，D（分）0.5—3

对于流量系统：P（% ）40--100，I （分）0.1—1

对于压力系统：P（% ）30--70 ，I（分）0.4—3

PID 控制器的原理是根据系统的被调量实测值与设定值之间的偏差，利用偏差的比例、积分、微分三个环节的不同组合计算出对广义被控对象的控制量。

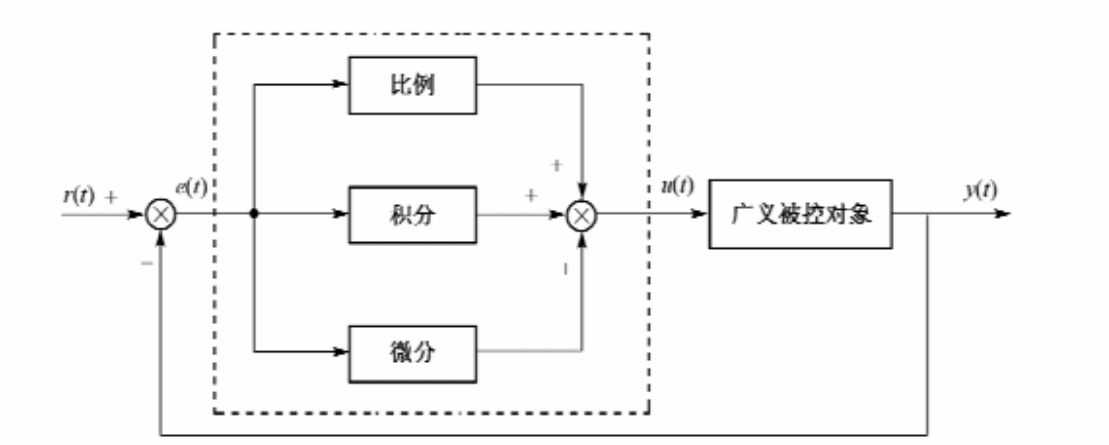


图 6.3 常规 PID 控制系统的原理框图

其中虚线框内的部分是 PID 控制器，其输入为设定值 r (t) 与被调量实测值 y (t)

构成的控制偏差信号 e(t) ：

$$e(t) = r(t) - y(t)$$

其输出为该偏差信号的比例、积分、微分的线性组合，也即 PID 控制律：式中 Kp 是比例常数，TI 为积分时间常数， TD 为微分时间常数。

$$u(t) = K_P\left[e(t) + \frac{1}{T_I}\int_0^t e(t)dt + T_D\frac{de(t)}{dt}\right]$$

一般来说在对三个系数进行调节时，遵循以下这些规律：

曲线振荡很频繁，比例度盘要放大；

曲线漂浮绕大湾，比例度盘往小扳；

曲线偏离回复慢，积分时间往下降；

曲线波动周期长，积分时间再加长；

曲线振荡频率快，先把微分降下来；

动差大来波动慢，微分时间应加长。

PID 控制算法是根据偏差的值，按照一定的函数关系进行计算，用所得的运算结果对系统进行控制。PID 控制有位置式控制和增量式控制，方程如下：

$$u_i = K\left[e_i + \frac{T}{T_I}\sum_{j=0}^{i} e_j + \frac{T_D}{T}(e_i - e_{i-1})\right] + u_0$$

$$\Delta u = K\left[e_i - e_{i-1} + \frac{T}{T_i}e_i + \frac{T_d}{T}(e_i - 2e_{i-1} + e_{i-2})\right]$$

在本智能车控制算法中，我们用位置式 PID 控制小车的横向运动，即小车的转角。对于位置式的 PID ，可以简化为以下式子：

U += （Kp * d_error + Ki * error +Kd*dd_error)

对于增量式的 PID, 也可以简化为以下式子：

U = Kp * error + Ki * Integral + Kd * derror

式中 Kp ，Ki ，Kd 分别是比例常数，积分常数，微分常数。

这种算法用来控制步进电机特别方便，对直流电机也可以采用，其实如果对控制有更高的要求或者干扰因素较多，可以对 PID 算法做各种改进，比如用梯形法做数值积分以提高精度，将差分改成一阶数字滤波等等。

6.3 舵机控制

舵机的控制决定了小车运动的轨迹，为了尽可能提高车模运行平均速度，针对不同的路面需要对运动轨迹进行优化。对于急转弯路面，应尽量靠内道行驶，而“S”形

弯应该取直穿过。为了简化问题，可以将小车的运动规律进行简化：车体近似沿着道路中心线运行时，改变舵机输出转角即可改变小车前轮转向，并改变车体与道路中心线的相对位置。在舵机控制上我们采用了位置式 PID 算法。采用简化公式

U +=（Kp * d_error + Ki * error +Kd*dd_error)

由于舵机本身相应就非常慢，而积分项会是系统的瞬时响应性能减弱，综合考虑之后，我们只用 PD 控制舵机。经过计算之后的均值和对应的舵机 PD 的参照角度上若采用一次线性关系，则对于本车模在低速的情况下，系统还是比较稳定的，但是提速就会存在来不及转急弯，直道上车身左右抖动，其原因有很多种，由于调整太频繁，或者舵机左右明显不对称造成的。在查找资料解决问题是，我们发现北京科技大学 CCD 一队曾经也存在这个问题，经过组员的研究和讨论之后，我们决定参考北京科技大学 CCD 一队技术报告上的，采用二次线性函数来控制舵机。即

a)微分项系数 Kd 则使用定值，原因是舵机在一般赛道中都需要好的动态响应能力；

b) 对 Kp，我们使用了二次函数的曲线，Kp 随黑线的偏离位置二次函数关系增大。

我们选择最后测试了一些 PID 参数，得到了较为理想的转向控制效果。有一定抗干扰和抗反光能力的黑线提取算法经过不断的试验之后，最后调出了适合是本系统的 PD 系数。函数结构如下：

KP=(X_PRESENT-VIDEO_CENTER)* (X_PRESENT-VIDEO_CENTER)/28+40;

steer=steer_center+(VIDEO_CENTER-X_PRESENT)*KP +KD*EE_PRESENT;

其中 KD 是常数，steer 是舵机输出量，VIDEO_CENTER 是视场中心，EE_PRESENT 是前后两次均值之差。

6.4 电机控制

在电机控制上我们采用 bang-bang 控制。因为 bang-bang 控制具有响应速度快、动态性能好的优点，在需要频繁加减速的智能车系统上应用具有很大优势，此外赛车驱动电机负载运行时，响应特性较软，而控制周期相对于电机响应特性较短，所以我们采用 Bang-Bang 控制方法来实现赛车速度的闭环控制。在每一个控制周期中，检测一次赛车的当前速度值。若速度值小于预定的速度值，则将驱动电机 PWM 输入的占空比置为 100% ；若速度大于预定的速度值，则将驱动电机 PWM 输入的占空比置为 0。

我们设定一速度，让赛车按此速度沿赛道匀速行驶。通过分析赛车实际的运行效果可以看出，速度最大下降幅度只有 19.8%，而且一般情况下，速度在平均值附近的波动幅度只有 3%。Bang-Bang 控制下的速度闭环可以很好达到速度控制的目的。 当小车在直线上行驶时，通过分析闭环速度的响应曲线，可以看出，速度从零增至稳态

速度的 70% 只需要 0.7 秒时间，达到稳态速度只需要 0.96 秒时间，远小于开环时速度达到稳态值的时间。速度流程图如图 6.4

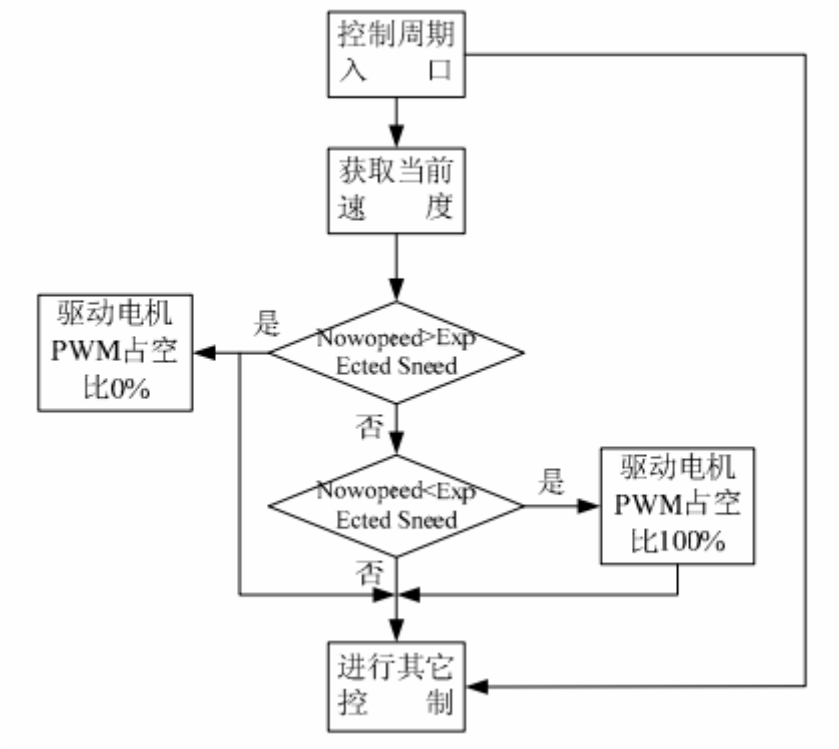


图 6.4 速度控制流程图

第七章　软件开发与调试

7.1 软件开发环境介绍

Codewarrior 是 Metrowerks 公司开发的软件集成开发环境，飞思卡尔所有系列的微控制器都可以在 codewarrior IDE 下进行软件开发。本模型车所用的处理器是 Mc9sXS128B，程序调试是在 codewarrior IDE 环境下实现的，所用语言为汇编。首先要新建一个基于 Mc9sXS128B 的 HCS12 的工程，选用语言为 C 语言，具体的过程如图 7-1 到 7-3 所示。建立好新的工程后，就可以在编译器里进行程序的编写。

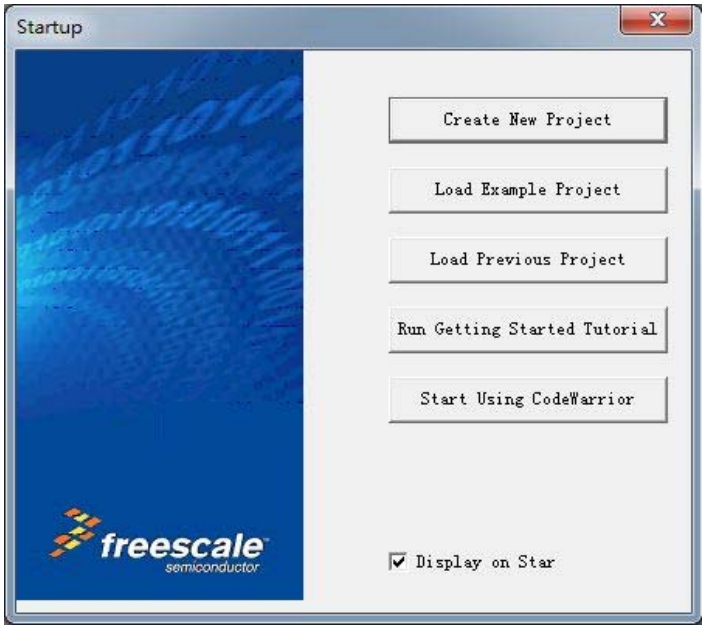


图 7-1　Codewarrioer 新建工程界面(a)

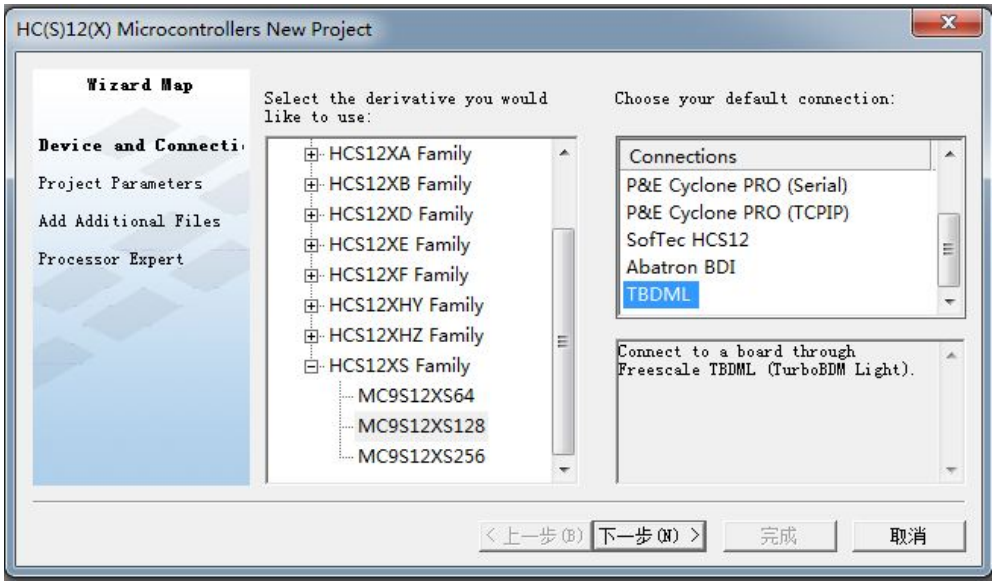


图 7-2 Codewarrioer 新建工程界面(b)

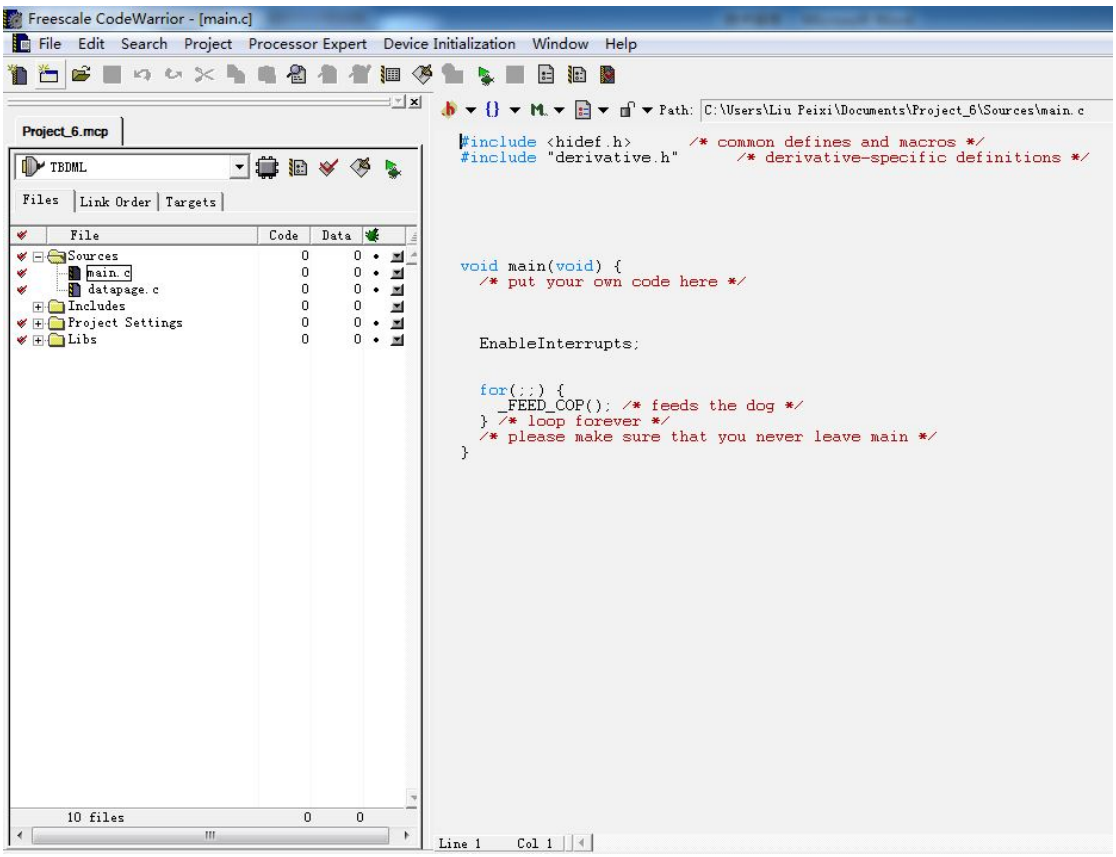


图 7-3 Codewarrioer 新建工程界面(c)

7.2 智能车整体的调试

在智能车的调试过程中，还常常需要考虑如下几点特殊的注意事项：

1) 外界环境的影响
2) 智能车质量
3) 智能车部件发热
4) 车轮
5) 避免过度转向
6) 提高稳定性
7) 提高转向伺服电机的反应速度
8) 算法设计

智能车设计的调试是应该注意：在智能车控制系统设计中，除了利用一般电子产品的调试方法之外，还可以设计一个专门的调试电路用于智能车的调整工作。调试电路一方面可以显示智能车控制电路的各种信息以及工作参数；另一方面还可以对工作参数进行现场修改。舵机调试：首先在程序里不断的修改舵机的控制量，确定舵机左转和右转极限的 PWM 值，记下该值留着在程序里设定左转和右转的极限值。然后用一小段跑道来测试舵机的转向，用不同的跑道段来测试模型车的转向是否符合要求，

当用的是直道时候要保证模型车的舵机位置绝对的居中，小“S”道时候，模型车给出的转角应该足够的小，大“S”时候转角应该近似为圆弧弦的角度，弯道时候要给足转角。

电机调试：像测试舵机一样用不同的赛道来测试电机给出的转速是否满足实际控制的需要，当用的是小“S”道时候，模型车应该保持直道上的速度，大“S”时候，应该适当的减速，以使得通过时候不会产生过大的超调，弯道时候要能够很快的将速度降下来，如果是急转弯时可以不给控制量或者让电机反转以达到刹车的效果。

整体调试：首先以一个较低的速度跑完整个赛道，然后再慢慢的提高速度，直到模型车在某一个地方出错，然后调整控制算法，如此反复，直到模型车能够以理想的速度，在理想的路线上运行完为止。

状态指示：在程序中使用开发板自带的 PB 灯作为速度指示装置。在赛车的整个运行过程中可以看到 PB 灯一直在闪烁。有时候程序或是 CS3020 出现了问题，PB 灯也是个很好的报警系统，因为这个时候 PB 灯的指示不正常，通常是全亮或是全灭。正常时，PB 灯会随着赛车的加减速灯亮的个数会有规律的变化。另外在程序中把 PB 灯用作了识别十字交叉的指示灯，即在过十字交叉的时候，PB 灯会全亮。

AD 参考电压的调试：本次设计中使用 LM336 稳压器的输出来作为 AD 的基准参考电压。考虑到不同的光线环境阈值的不同以及更改阈值的便捷性，在这一模块电路中使用一个 5K 的滑动变阻器来扩大可选的 AD 基准电压的范围。实验证明，5K 电位器在本模块电路中的可调电压范围为 0—5V, 完全可以满足要求。 经过大量实验，AD 基准电压定为 2.7V 。因为在这个参考电压下，AD 模块能最大限度地区分出黑白点，而且在试行的过程中不会发生意外的抖动而冲出赛道。在实验中还发现，如果这个参考电压偏离选定的 AD 基准电压超过 0.2V 时，就会影响到采集图像的准确度。调试时发现，近处几行的图像受到的影响相对较小，基本上可以如实反映出黑白点的变化，但是中远几行就会出现比较明显的错误，特别是最远处的几行，有些地方甚至黑白不分。从理论上讲，如果 AD 参考电压不适当，会出现两种情况：一是 AD 参考电压过大，导致黑白点的阈值相差不多，即黑白点没有明显的差别，这样此块区域就可能被识别成全白或是全黑。还有一种情况是 AD 参考电压过小(不能低于视频信号的电压峰值，否则会造成转换失败而得出错误的结果) ，这样黑白点的阈值相差太大。这对于已经调试好的程序来说，会把某些白点当成黑点，或是把某些黑点当成白点，而且这样很易受光线的影响。

7.3 辅助开发工具

利用 Matlab 良好的图像处理功能，与串口相结合对智能车系统进行调试。串口主要是读取并捕获单片机中存储的信息，然后用 VB 编程的小软件回显图像，这样有利于分析图像信息。其小软件界面如下：

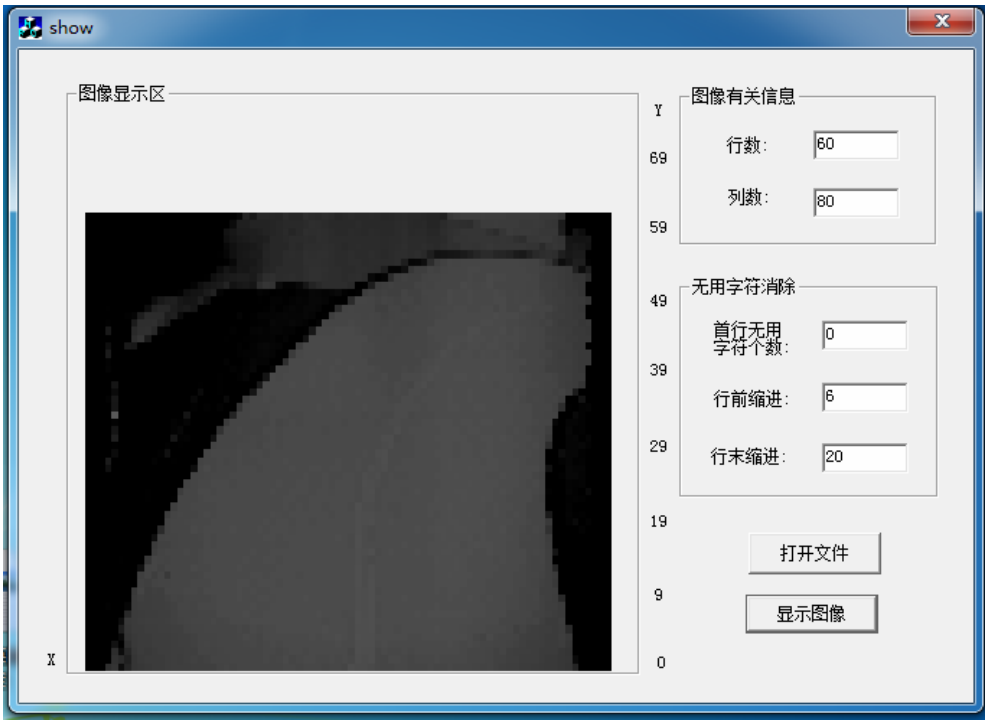


图 7.4 显示图像软件截图

第八章　总结与展望

8.1 总结

经过队员们几个月的努力，我们完成了智能车系统的制作。整个车模硬件电路简单，在 CCD 传感器的控制下，通过判断，使模型车转向准确稳定，能够完全通过各种弯道和十字交叉路口。该模型车在满足大赛要求的前提下，具有良好的自主道路识别能力和稳定性，并能够以较快的速度行驶。 在最初的方案选择阶段，通过比较不同方法可能得到的结果明确了我们研究的方向。

在机械设计方面，我们对模型车的性能不断的探索改进了前轮定位、差速、舵机力矩等，使得舵机的转向更加灵活、更容易控制，打滑现象基本解决。在控制算法方面，我们通过建模和仿真，最终确定使用模糊控制算法对模型车的方向和速度进行控制。在图像采集和图像处理方面，我们以 S12 单片机为核心，结合摄像头的安装设计了合理的采集方法，并对采集的图像用阈值二值化方法进行处理，降低干扰。

8.2 展望

同时由于时间的不足，在模型车的研究中还有许多需要改进的地方：

1. 本文所建立的智能汽车的运动模型是在理想情况下建立的，其实际情况要复杂的多，所建立的模型过于简单。另一方面，智能汽车是一个高度非线性化、时变的系统，如何对模型中的参数进行估计以及使模型更能反映实际车辆运动是一个需要进一步解决的问题。

2. 模糊控制对数学模型很难建立的系统具有先天性的优势，而在本文中只是用一种很简单的模糊控制方法来控制模型车，考虑的模型车的其他干扰因素还不够多，因此如何进一步改进控制算法是十分有必要的。

3. 在本模型车在处理算法中，处理的频率为 50Hz，模型车的处理算法的频率，极大地限制着模型车整体速度的提高，在以后的研究中，我们将考虑使用非标准的频率高的特殊摄像头或者在算法中实现处理频率的提高。

参考文献

[1] 卓晴, 黄开胜,邵贝贝. 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] 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,2004.

[3] 周继明，江世明，彭解华. 传感器技术与应用[M]. 长沙：中南大学出版社,2005.

[4] Ramon Pallas-Areny, John G. Webster. 传感器和信号调节(第二版)[M].北京：清华大学出版社,2003.

[5] 宗光华. 机器人的创意与实践[M]. 北京：北京航空航天大学出版社,2004.

[6] 卓晴，王珽，王磊. 基于面阵 CCD 的赛道参数检测算法.

[7] MC33886 Technical Data, Freescale Semiconductor July 2005.

[8] 余志生. 汽车理论（第 3 版）[M]. 机械工业出版社,2004.

[9] LM1117 800mA Low-Dropout Linear Regulator, National Semiconductor February 1999.

[10] MC34063A/E DC-DC Converter Control Circuits, Motorola.

[11] Motorola MC9S12DG128 Device User Guide Motorola, Inc.

[12] LM1881 Video Sync Separator, National Semiconductor June 2003.

[13] 熊万龙. 基于 Kalman 滤波器的自主车辆定位方法研究[J].湖南：国防科技大学学位论文，2005.

[14] 殷人昆，陶永雷，谢若阳，盛绚华. 数据结构(用面向对象方法和 C++ 描述) . 北京：清华大学出版社.2005.

[15] 关柏青. 基于彩色视觉和模糊控制的移动机器人路径跟踪[J]. 河北工业大学学位论文.2002.

[16] 谢剑英，贾青. 微型计算机控制技术[M]. 北京：国防工业出版社，2004.

[17] 飞思卡尔 HC08/HCS12 系列微控制器开发环境- Codewarrior 使用指南.

[18] 孙景琪等，视频技术与应用. 北京工业大学出版社.

[19] 刘叔军，盖晓华，樊京，崔世林等.MATLAB 7.0 控制系统应用与实例[M]. 北京. 机械工业出版社.2006.

附录 A

模型车技术参数统计

项　　目	参　　数
路径检测方法（赛题组）	摄像头
车模几何尺寸（长、宽、高）（毫米）	250*180*205
车模轴距/ 轮距（毫米）	172/180
车模平均电流（匀速行驶）(毫安)	190
电路电容总量（微法）	1400
传感器种类及个数	编码器 1 个，摄像头 1 个
新增加伺服电机个数	0
赛道信息检测空间精度（毫米）	10
赛道信息检测频率（次/ 秒）	50
主要集成电路种类/ 数量	最小系统板 1 片、信号板和驱动板集成为 1 片主板
车模重量（带有电池）（千克）	1

附录 B

源程序

```
#include <hidef.h>          /* common defines and macros */
#include <MC9S12XS128.h>        /* derivative information */
#pragma LINK_INFO DERIVATIVE "mc9s12xs128"
#define uchar unsigned char
#define uint unsigned int

#define rightturn 1
#define leftturn 2
#define distortion 3
uchar pic[70][80];
uchar gpline;
uchar gpflag;
uchar gppoint;
unsigned int count;
int value;
```

```
uint row;
uint line;
uchar d_left_flag;
uchar s_left_flag;
uchar d_right_flag;
uchar s_right_flag;
uchar find_flag;
uchar l_startline;
uchar r_startline;
uchar l_limit;
uchar r_limit;
uint black_line_left[70];
uint black_line_right[70];
uint black_line_center[70];
uint black_line_width[70];
uint preblack_line_left;
uint preblack_line_right;
uint preblack_line_center;
uint preblack_line_row;
uint preblack_line_rrow;
uint preblack_line_lrow;
uint STEER;
uint preSTEER;
uint control_line;
uint control_row;
uchar control_flag;
struct line
{
  uint brow;
  uint bline;
  uint erow;
  uint eline;
}leftline1,rightline1,leftline2,rightline2;
uchar leftlineflag1;
uchar rightlineflag1;
uchar leftlineflag2;
uchar rightlineflag2;
uint lefthuiduhe;
uint righthuiduhe;
uchar righthui[60];
uchar lefthui[60];
uint lefthuidu;
uint righthuidu;
uchar roadshape;
```

```
uchar left_shape;
uchar right_shape;
int left_cha[60];
int right_cha[60];
uint left_guai;
uint right_guai;
uchar left_a;
uchar left_b;
uchar left_c;
uchar left_d;
uchar right_a;
uchar right_b;
uchar right_c;
uchar right_d;
uint _count1;
uint _count2;
int _state[60];
uint state;
uint _max;
uint _min;
uint _length;
uchar _left;
uchar _right;
uint precontrol_line;
unsigned int er;
uchar cross_flag;
uchar crossflag;
uchar crossflag_left;
uchar crossflag_right;
uint speed_now;
int En3_L;
int En2_L;
int En1_L;
int temp;
int s;
uchar flag_1;
uchar flag_2;
uchar flag_3;
uchar flag_4;
uchar flag_5;
uchar flag_6;
uchar start_left_flag;
uchar start_right_flag;
uchar start_flag;
```

```
uint start_left_row[2];
uint start_right_row[2];
uint start_left_line[2];
uint start_right_line[2];
uint VALVE1;
uchar stop_flag;
uchar straight_flag;
uint count1;
uint count2;
uchar z;
uint end_row;
uint end_line;
uint last_row;
uint _left_flag;
uint _left_count;
uint _right_flag;
uint _right_count;
uint stop_time;
uint near_row,near_line,near_flag,far_row,far_line,far_flag;
float l_k;
float r_k;
uint SPEED;
uchar t;
const M_choice[]={20,20,20,20,20,19,19,19,19,19,
                  18,18,18,18,18,17,17,17,17,17,
                  16,16,16,16,16,15,15,15,15,15,
                  14,14,14,14,14,13,13,13,13,13,
                  12,12,12,12,12,11,11,11,11,11,
                  10,10,10,10,10,10,10,10,10,10,
                  10,10,10,10,10,10,10,10,10,10
                  };
const J_choice[]={35,35,34,34,33,33,32,32,31,31,
                  30,30,29,29,28,28,27,27,26,26,
                  25,25,24,24,23,23,23,23,23,23,
                  22,22,22,22,22,22,22,22,22,22,
                  21,21,20,20,19,19,18,18,17,17,
                  16,15,14,13,12,11,10,10,10,10,
                  10,10,10,10,10,10,10,10,10,10
                  };
const K_choice[]={49,49,48,48,47,47,47,45,44,43,
                  41,41,39,38,36,35,35,35,34,33,
                  33,32,32,32,31,30,30,30,29,29,
                  29,29,28,28,28,27,27,26,26,25,
                  25,24,24,23,23,22,22,21,21,20,
```

```
                        20,19,19,18,18,18,17,17,16,15
                       };
#define steercenter 5045
#define steerright 4500
#define steerleft 5550
#define centerline 39

//////////////////////////////////
void init_function1()
{
   gpline=0;                                    //一场中行计数清零
   gpflag=0;                                    //清除场采集结束标志
   gppoint=0;
   value=30;
   count=0;
   row=0;
   line=0;
   control_line=0;
   control_row=0;
   d_left_flag=0;
   d_right_flag=0;
   s_left_flag=0;
   s_right_flag=0;
   find_flag=0;
   l_startline=4;
   r_startline=75;
   preblack_line_left=0;
   preblack_line_right=0;
   En3_L=0;
   En2_L=0;
   En1_L=0;
   temp=0;
   s=0;
   z=0;
   stop_time=0;
   stop_flag=0;
   start_flag=0;
   for(row=0;row<70;row++)
   {
      black_line_left[row]=0;
      black_line_right[row]=0;
      black_line_center[row]=0;
      black_line_width[row]=0;
   }
```

```
}

void init_function()
{
  /*********************************************************************/
  //时钟初始化
  /*********************************************************************/
    CLKSEL=0X00;       //disengage PLL to system
    PLLCTL_PLLON=1;     //turn on PLL

    SYNR=0xc0|0x05;        //这里有问题

    REFDV=0x80|0x01;              //pllclock=2*osc*(1+SYNR)/(1+REFDV)=48MHz;

    POSTDIV=0x00;                    //pllclock=2*osc*(1+SYNR)/(1+REFDV)=128MHz;
    _asm(nop);              //BUS CLOCK=16M
    _asm(nop);

    while(!(CRGFLG_LOCK==1));     //when pll is steady ,then use it;
    CLKSEL_PLLSEL =1;              //engage PLL to system;
    COPCTL = 0X00; //关闭看门狗
    DisableInterrupts;
  /*********************************************************************/
  //I/O 口初始化
  /*********************************************************************/
  DDRA=0xFF;                                //
  DDRT=0x00;                               //测速接口 PT7 设置输入
  DDRH=0x00;                               //场同步接口 PH1 设置输入
  //DDRB=0x00;                               //调试灯 B 口设置输出
  DDRJ=0x00;
  DDRK=0x00;
  /*********************************************************************/
  // ECT Initialize
  /*********************************************************************/
  TSCR1_TFFCA=1; //快速清除 flag：PAOVF PAIF
  TIOS=0x1F;                              //0-4 通道设置为输出比较，5-7 通道设置为输入捕捉
  TIE=0;                                  //初始时屏蔽所有通道的中断
  TSCR2=0x04;                                //溢出中断，计时器频率由总线频率 16 分频得到
==3.125MHz
  TCTL3=0x50;                             //检测 PT6.7 上升沿
  TSCR1=0x80;                             //启动 ECT 模块及计数器
  /*
  PACTL_PAMOD=0; //事件计数方式
  PACTL_PAEN=1; //启动脉冲累加器 A
```

```
  PACNT = 0x0000;
  */
  IRQCR=0;                              //关闭行同步中断(IRQ)

  IRQCR_IRQEN =0;                       //关硬件外部中断，行中断

  ////////////////////////////////////////////////////////////////

  ATD0CTL1=0x00;    //7:1-外部触发,65:00-8 位精度,4:放电,3210:ch
  ATD0CTL2=0x40;    //禁止外部触发, 中断禁止
  ATD0CTL3=0x08;    //左对齐无符号,每次转换 1 个序列, No FIFO, Freeze 模式下继续转
  ATD0CTL4=0x00;    //765:采样时间为 4 个 AD 时钟周期,ATDClock=[BusClock*0.5]/[PRS+1]
  ATD0CTL5=0x00;
  ATD0DIEN=0x00;    //禁止数字输入

  ////////////////////////////////////////////////////////////////
  PWMCTL=0xE0;                        //将 PWM4 和 PWM5 合成 16 位，PWM01 不合并，PWM23
合并，PWM67 合并
    PWMPRCLK=0x33;                      //时钟 A=时钟 B=bus clock/8=50M/8=6.25M
    PWMSCLA=0x01;                       //时钟 SA=时钟 A/2/1=3.125M
    PWMSCLB=0x01;                       //时钟 SB=时钟 B/2/1=3.125M
    PWMCLK=0xF3;//0xC3;                 //时钟 A 控制 PWM4 和 PWM5，B 控制 PWM2 和 PWM3，SA
控制 PWM0 和 PWM1，SB 控制 PWM6 和 PWM7
    PWMPOL=0xFF;                        //所有通道位极限为 1
    PWMCAE=0x00;                        //所有通道左对齐
    PWMPER0=250;                        //电机反转频率为 3.125MHz/312=10KHz
    PWMDTY0=0;
    PWMPER1=250;
    PWMDTY1=0;                          //电机反转初值赋 0
    PWMPER67=62500;
    PWMDTY67=steercenter;
    PWMPER23=625;                       //电机正转频率为 6.25MHz/625=10KHz
    PWMDTY45=0;                         //电机正转初值赋 0
    PWMPER45=625;                       //舵机频率为 3.125MHz/62500=50Hz
    PWMDTY23=0;                         //舵机初值居中
    PWME=0xFF;                          //启动 PWM

        }


void delay1(unsigned int m,unsigned int n)
 {
  unsigned int i,j;
  for(i=0;i<m;i++)
```

```
   {
      for(j=0;j<n;j++);
   }
  }

void delay_1ms(unsigned int m)
{
   unsigned int tt,rr;
   tt=0;
   rr=0;
   for(rr=0;rr<m;rr++)
      {
         for(tt=0;tt<2140;tt++);
      }
}

unsigned int fabs1(int a,int b)
{
   int dif;
   if(a>b)
      {
         dif=(a-b);
      }
   else
   if(b>a)
      {
         dif=(b-a);
      }
   else
   if(b==a)
      {
         dif=0;
      }
   return dif;
}
void blacklineget()//search_right();line_center(); detect_right();line_center();
{
   _left=0,_right=0;
   preblack_line_left=0,preblack_line_right=0,preblack_line_center=0;
   for(row=2;row<60;row++)
      {
         d_right_flag=0,s_right_flag=0,d_left_flag=0,d_right_flag=0;
         if(_left<1)
            {
```

```
            search_left();
            if(black_line_left[row]>5&&black_line_left[row]<76)
              {
                _left++;
              }
          }
        else
          {
            detect_left();
            if(black_line_left[row]>5&&black_line_left[row]<76)
              {
                _left++;
              }
          }
        if(_right<1)
          {
            search_right();
            if(black_line_right[row]>5&&black_line_right[row]<76)
              {
                _right++;
              }
          }
        else
          {
            detect_right();
            if(black_line_right[row]>5&&black_line_right[row]<76)
              {
                _right++;
              }
          }
      }
          for(row=2;row<60;row++)
      {
        line_center();
      }
}

void Cross()
{
   uint _leftline1=0,_rightline1=0;
   uint left_max=0,right_max=0;
   uchar left_=0,right_=0;
   float l_near_row=0,l_near_line=0,l_far_row=0,l_far_line=0,l_row=0;
   float r_near_row=0,r_near_line=0,r_far_row=0,r_far_line=0,r_row=0;
```

```
left_a=0,left_b=0,left_c=0,left_d=0,right_a=0,right_b=0,right_c=0,right_d=0;
leftline1.brow=2,leftline1.bline=black_line_left[2],leftline1.erow=0,leftline1.eline=0;
rightline1.brow=2,rightline1.bline=black_line_right[2],rightline1.erow=0,rightline1.eline=0;
leftline2.brow=0,leftline2.bline=0,leftline2.erow=0,leftline2.eline=0;
rightline2.brow=0,rightline2.bline=0,rightline2.erow=0,rightline2.eline=0;
leftlineflag1=0,rightlineflag1=0;
leftlineflag2=0,rightlineflag2=0;
lefthuiduhe=0,righthuiduhe=0,lefthuidu=0,righthuidu=0;
roadshape=0,right_shape=0,left_shape=0;
left_guai=0,right_guai=0;
PORTA=0X00;
for(row=0;row<60;row++)
{
    left_cha[row]=0;
    right_cha[row]=0;
    lefthui[row]=0;
    righthui[row]=0;
}
//找到两段直线的起点和终点
for(row=2;row<=60;row++)
{
  if(black_line_left[row]==0)
  {
     if(black_line_left[row-1]>5&&black_line_left[row-1]<75)
     {
         leftline1.erow=row-1;
         leftline1.eline=black_line_left[row-1];
         leftlineflag1=1;
         break;
     }
  }
 }
 if(leftline1.erow<59)
 {
     for(row=leftline1.erow+1;row<60;row++)
     {
        if(black_line_left[row]>5&&black_line_left[row]<75)
        {
           leftline2.brow=row;
           leftline2.bline=black_line_left[row];
           break;
        }
     }
 }
```

```
for(row=leftline2.brow;row<=60&&row>2;row++)
{
   if(black_line_left[row]==0)
   {
      if(black_line_left[row-1]>5&&black_line_left[row-1]<75)
      {
         leftline2.erow=row-1;
         leftline2.eline=black_line_left[row-1];
         leftlineflag2=1;
         break;
      }
   }
}

for(row=2;row<=60;row++)
{
  if(black_line_right[row]==0)
  {
     if(black_line_right[row-1]>5&&black_line_right[row-1]<75)
     {
         rightline1.erow=row-1;
         rightline1.eline=black_line_right[row-1];
         rightlineflag1=1;
         break;
     }
  }
}
for(row=rightline1.erow+1;row<=60;row++)
{
   if(black_line_right[row]>5&&black_line_right[row]<75)
   {
      rightline2.brow=row;
      rightline2.bline=black_line_right[row];
      break;
   }
}
for(row=rightline2.brow;row<=60&&row>2;row++)
{
   if(black_line_right[row]==0)
   {
      if(black_line_right[row-1]>5&&black_line_right[row-1]<75)
      {
         rightline2.erow=row-1;
         rightline2.eline=black_line_right[row-1];
```

```
            rightlineflag2=1;
            break;
        }
    }
 }
//道路修正
if(leftlineflag2==1)
{
    for(row=leftline1.erow+1;row<leftline2.brow;row++)
    {
        if(black_line_left[row]==0)
        {
            left_c++;
        }
    }
   if(left_c>=25)
   {
      leftlineflag2=0;
      for(row=leftline2.brow;row<60;row++)
      {
          black_line_left[row]=0;
      }
   }
}
if(rightlineflag2==1)
{
    for(row=rightline1.erow+1;row<rightline2.brow;row++)
    {
        if(black_line_right[row]==0)
        {
            right_c++;
        }
    }
   if(right_c>=25)
   {
      rightlineflag2=0;
      for(row=leftline2.brow;row<60;row++)
      {
          black_line_left[row]=0;
      }
   }
}
//找到第一段直线的拐点
if(leftlineflag1==1)
```

```
  {
    for(row=3;row<=leftline1.erow-1;row++)
    {
        left_cha[row]=black_line_left[row-1]-black_line_left[row];
        if(left_max<fabs1(left_cha[row],0))
        {
            left_max=fabs1(left_cha[row],0);
            if(left_max>2)
            {
               left_guai=row-2;
            }
        }
    }
    if(left_guai==0)
    {
        for(row=3;row<=leftline1.erow-2;row++)
        {

if(left_cha[row]<0&&(left_cha[row+1]<0||left_cha[row+1]==0)&&(left_cha[row+2]<0||left_cha[row+2]
==0)&&(left_cha[row+3]<0||left_cha[row]==0))
            {
               left_guai=row-2;
               break;
            }
        }
    }
  }
  if(rightlineflag1==1)
  {
    for(row=3;row<=rightline1.erow-1;row++)
    {
        right_cha[row]=black_line_right[row]-black_line_right[row-1];
        if(right_max<fabs1(right_cha[row],0))
        {
            right_max=fabs1(right_cha[row],0);
            if(right_max>2)
            {
               right_guai=row-2;

            }
        }
    }
    if(right_guai==0)
    {
```

```
            for(row=3;row<=rightline1.erow-2;row++)
            {

if(right_cha[row]<0&&(right_cha[row+1]<0||right_cha[row+1]==0)&&(right_cha[row+2]<0||right_cha[row+2]==0)&&(right_cha[row+3]<0||right_cha[row]==0))
                {
                    right_guai=row-2;
                    break;
                }
            }
        }
    }

    //用拐点替代第一段直线的终点
    if(right_guai>2)
    {
        rightline1.erow=right_guai;
        rightline1.eline=black_line_right[rightline1.erow];
    }
    if(left_guai>2)
    {
        leftline1.erow=left_guai;
        leftline1.eline=black_line_left[leftline1.erow];
    }

     //再次道路优化
     if(leftline2.brow>55)
     {
          leftlineflag2=0;
     }
     if(rightline2.brow>55)
     {
          rightlineflag2=0;
     }
    //计算第一段直线的终点到第二段直线的起点的近似平均灰度值
     if(leftlineflag1==1&&leftlineflag2==1)
     {
        _leftline1=(leftline1.eline+leftline2.bline)/2;
        for(row=leftline1.erow+1;row<leftline2.brow;row++)
        {
            lefthuiduhe=lefthuiduhe+pic[row][_leftline1];
            lefthui[row]=pic[row][_leftline1];
            if(lefthui[row]>0)
            {
```

```
                if(lefthui[row]>60)
                {
                    left_a++;
                }
                  else
                if(lefthui[row]<60)
                {
                    left_b++;
                }
        }
    }
    if(leftline2.brow-leftline1.erow>1)
    {
        lefthuidu=lefthuiduhe/(leftline2.brow-leftline1.erow-1);
    }
}



if(rightlineflag1==1&&rightlineflag2==1)
{
  _rightline1=(rightline1.eline+rightline2.bline)/2;
  for(row=rightline1.erow+1;row<rightline2.brow;row++)
    {
       righthuiduhe=righthuiduhe+pic[row][_rightline1];
       righthui[row]=pic[row][_rightline1];
       if(righthui[row]>0)
       {
          if(righthui[row]>60)
          {
              right_a++;
          }
          if(righthui[row]<60)
          {
              right_b++;
          }
       }
    }
  if(rightline2.brow-rightline1.erow>1)
  {
     righthuidu=righthuiduhe/(rightline2.brow-rightline1.erow-1);
  }
}
```

```
    //再次对道路进行修正
    if(left_a>=25)
    {
          leftlineflag2=0;
    }
    if(right_a>=25)
    {
          rightlineflag2=0;
    }

//第一段直线左右拐的判断
  if(leftline1.erow>leftline1.brow&&leftline1.eline>leftline1.bline)
  {
     left_shape=leftturn;
  }
  if(rightline1.erow>rightline1.brow&&rightline1.eline>rightline1.bline)
  {
     right_shape=leftturn;
  }

  if(leftline1.erow>leftline1.brow&&leftline1.eline<leftline1.bline)
  {
     left_shape=rightturn;
  }
  if(rightline1.erow>rightline1.brow&&rightline1.eline<rightline1.bline)
  {
     right_shape=rightturn;
  }

//十字叉的判断
if(leftlineflag2==1&&rightlineflag2==0)
{
    if(right_guai>3&&lefthuidu>=50&&left_a>=10)
    {
        roadshape=distortion;
    }
} else
if(leftlineflag2==0&&rightlineflag2==1)
{
    if(left_guai>3&&righthuidu>=60&&right_a>=10)
    {
        roadshape=distortion;
    }
```

```
 }else
 if(leftlineflag2==1&&rightlineflag2==1)
 {

if((righthuidu>=60&&lefthuidu>=50)||(right_shape==leftturn||right_shape==0)&&(left_shape==righttur
n||left_shape==0)||(left_b<6&&left_a<20)||(left_b==0&&left_a>=20))
    {
        roadshape=distortion;
    }
 }
 if(roadshape==distortion)
 {
      PORTA=0X0F;
 }

 if(roadshape==distortion)
 {
   if(leftlineflag1==1)
   {
      if(leftline1.erow>20)
      {
         l_near_row=2,l_near_line=black_line_left[2];
         l_far_row=leftline1.erow-3,l_far_line=black_line_left[leftline1.erow-3];
         l_k=(l_far_line-l_near_line)/(l_far_row-l_near_row);
         for(row=l_near_row+1;row<60;row++)
         {
             l_row=row;
             black_line_left[row]=(int)(l_k*(l_row-l_near_row-1)+l_near_line);
         }
      }
      else
      {
          if(leftlineflag2==1)
          {
              l_far_row=leftline2.brow+2,l_far_line=black_line_left[leftline2.brow+2];
              if(leftline1.erow>4)
              {
                 l_near_row=leftline1.erow-3,l_near_line=black_line_left[leftline1.erow-3];
              }
              else
              {
                 l_near_row=leftline1.erow,l_near_line=black_line_left[leftline1.erow];
              }
              l_k=(l_far_line-l_near_line)/(l_far_row-l_near_row);
```

```
            for(row=l_near_row+1;row<60;row++)
            {
               l_row=row;
               black_line_left[row]=(int)(l_k*(l_row-l_near_row-1)+l_near_line);
            }
        }
        else
        {
            l_near_row=leftline1.brow,l_far_line=black_line_left[leftline1.brow];
            if(leftline1.erow>4)
            {
               l_far_row=leftline1.erow-3,l_far_line=black_line_left[leftline1.erow-3];
            }
            else
            {
               l_far_row=leftline1.erow,l_far_line=black_line_left[leftline1.erow];
            }
            l_k=(l_far_line-l_near_line)/(l_far_row-l_near_row);
            for(row=l_near_row+1;row<60;row++)
            {
               l_row=row;
               black_line_left[row]=(int)(l_k*(l_row-l_near_row-1)+l_near_line);
            }
          }
        }
    }
if(rightlineflag1==1)
{
    if(rightline1.erow>20)
    {
       r_near_row=2,r_near_line=black_line_right[2];
       r_far_row=rightline1.erow-3,r_far_line=black_line_right[rightline1.erow-3];
       r_k=(r_far_line-r_near_line)/(r_far_row-r_near_row);
       for(row=r_near_row+1;row<60;row++)
       {
           r_row=row;
           black_line_right[row]=(int)(r_k*(r_row-r_near_row-1)+r_near_line);
       }
    }
    else
    {
        if(rightlineflag2==1)
        {
            r_far_row=rightline2.brow+2,r_far_line=black_line_right[rightline2.brow+2];
```

```
                if(rightline1.erow>4)
                {
                   r_near_row=rightline1.erow-3,r_near_line=black_line_right[rightline1.erow-3];
                }
                else
                {
                   r_near_row=rightline1.erow,r_near_line=black_line_right[rightline1.erow];
                }
                r_k=(r_far_line-r_near_line)/(r_far_row-r_near_row);
                for(row=r_near_row+1;row<60;row++)
                {
                   r_row=row;
                   black_line_right[row]=(int)(r_k*(r_row-r_near_row-1)+r_near_line);
                }
            }
            else
            {
                r_near_row=rightline1.brow,r_far_line=black_line_right[rightline1.brow];
                if(rightline1.erow>4)
                {
                   r_far_row=rightline1.erow-3,r_far_line=black_line_right[rightline1.erow-3];
                }
                else
                {
                   r_far_row=rightline1.erow,r_far_line=black_line_right[rightline1.erow];
                }
                r_k=(r_far_line-r_near_line)/(r_far_row-r_near_row);
                for(row=r_near_row+1;row<60;row++)
                {
                   r_row=row;
                   black_line_right[row]=(int)(r_k*(r_row-r_near_row-1)+r_near_line);
                }
              }
            }
       }

    for(row=2;row<60;row++)
    {

if(black_line_left[row]>5&&black_line_left[row]<75&&black_line_right[row]>5&&black_line_right[row]<
75)
      {
         if(black_line_left[row]>black_line_right[row])
            {
```

```
            black_line_center[row]=(black_line_left[row]+black_line_right[row])/2;
          }
      }
     }

 }
}
```

void start_line() //先在已知黑线两边一定区域内搜索起跑线或十指交叉标志横线，若找到则进一步在此标志附近区域进行精确判定，否则跳出

```
{
   uint i;
   start_left_flag=0;
   start_right_flag=0;
   //start_flag=0;
   start_left_row[0]=0;
   start_left_row[1]=0;
   start_right_row[0]=0;
   start_right_row[1]=0;
   start_left_line[0]=0;
   start_left_line[1]=0;
   start_right_line[0]=0;
   start_right_line[1]=0;
   flag_1=0;
   flag_2=0;
   flag_3=0;
   flag_4=0;
   flag_5=0;
   flag_6=0;
   VALVE1=40;
   for(row=6;row<50;row++)
     {
        if(black_line_center[row-1]>20&&black_line_center[row-1]<60)
         {
           /*if(black_line_left[row-1]<5||black_line_left[row-1]>72)
             {
               black_line_left[row-1]=black_line_center[row-1]+2;
             }*/
           for(line=black_line_center[row-1]+3;line<black_line_center[row-1]+8;line++)
             {
               if(pic[row][line]<VALVE1&&start_left_flag<2)
                 {
                   start_left_row[start_left_flag]=row;
                   start_left_line[start_left_flag]=line+3;
```

```
                    start_left_flag++;
                }
            }
          /*if(black_line_right[row-1]<5||black_line_right[row-1]>72)
            {
              black_line_right[row-1]=black_line_center[row-1]-2;
            }*/
          for(line=black_line_center[row-1]-3;line>black_line_center[row-1]-8;line--)
            {
              if(pic[row][line]<VALVE1&&start_right_flag<2)
                {
                  start_right_row[start_right_flag]=row;
                  start_right_line[start_right_flag]=line+3;
                  start_right_flag++;
                }
            }
        }
    }
if(start_right_flag>0&&start_left_flag>0)
    {
       for(i=0;i<2;i++)
        {
          for(row=start_right_row[i]-2;row<start_right_row[i]+5;row++)
            {
              if(black_line_center[row-1]>20&&black_line_center[row-1]<60)
                {
                  for(line=black_line_center[row-1];line<black_line_center[row-1]+3;line++)
                    {
                      if(pic[row][line]<VALVE1)
                        {
                          flag_1++;
                        }
                    }
                  for(line=black_line_center[row-1];line>black_line_center[row-1]-3;line--)
                    {
                      if(pic[row][line]<VALVE1)
                        {
                          flag_2++;
                        }
                    }
                }
            }
        }
      if(flag_1<3&&flag_2<3)
```

```
{
    for(i=0;i<2;i++)
     {
        for(row=start_right_row[i]-2;row<start_right_row[i]+5;row++)
          {
            if(black_line_center[row-1]>20&&black_line_center[row-1]<60)
              {
                for(line=black_line_center[row-1]+2;line<black_line_center[row-1]+8;line++)
                  {
                    if(pic[row][line]<VALVE1)
                      {
                        flag_3++;
                      }
                  }
                for(line=black_line_center[row-1]-2;line>black_line_center[row-1]-8;line--)
                  {
                    if(pic[row][line]<VALVE1)
                      {
                        flag_4++;
                      }
                  }
              }
          }
        for(row=start_left_row[i]-2;row<start_left_row[i]+5;row++)
          {
            if(black_line_center[row-1]>20&&black_line_center[row-1]<60)
              {
                for(line=black_line_center[row-1]+2;line<black_line_center[row-1]+8;line++)
                  {
                    if(pic[row][line]<VALVE1)
                      {
                        flag_3++;
                      }
                  }
                for(line=black_line_center[row-1]-2;line>black_line_center[row-1]-8;line--)
                  {
                    if(pic[row][line]<VALVE1)
                      {
                        flag_4++;
                      }
                  }
              }
          }
     }
```

```
            if(flag_3>10&&flag_4>10)
              {
                for(row=start_left_row[i]-2;row<start_left_row[i]+5;row++)
                  {
                      for(line=64;line<70;line++)
                            {
                              if(pic[row][line]<VALVE1)
                                {
                                  flag_5++;
                                }
                            }
                          for(line=10;line>4;line--)
                            {
                              if(pic[row][line]<VALVE1)
                                {
                                  flag_6++;
                                }
                            }
                  }
                start_flag=1;
                PORTA=0xf0;
                stop_flag=1;
                z++;
                if(z==1)
                  {
                      //stop_time=count;
                  }
              }
          }
      }
}



void steerprotect()
{
STEER=steercenter+(M_choice[control_row]+20)*(control_line-
centerline);//M_choice[control_row]+(int)S_I*(control_line-precontrol_line)
   if(STEER<=steerright-20){STEER=steerright;}
   else
   if(STEER>=steerleft+20){STEER=steerleft;}

   PWMDTY67=STEER;
   preSTEER=STEER;
```

```
}

void speedadjust()
{
   unsigned int SPEED_P=35,SPEED_D=15;
   s=0;
   En1_L=SPEED-speed_now;
   if(PWMDTY23>500||PWMDTY23<0)
     {
        PWMDTY23=temp;
     }
   temp=PWMDTY23;
   if(fabs1(En1_L,0)>=50)
     {
        if(En1_L>0)
          {
             PWMDTY23=550;
          }
        else
          {
             PWMDTY23=20;
          }
     }
   else
   if(speed_now>=SPEED)
    {
            PWMDTY45=0;
            if((0-En1_L)<10)
             {
                SPEED_P=20;
                SPEED_D=10;
             }
            else
            if((0-En1_L)<20)
             {
                SPEED_P=25;
                SPEED_D=15;
             }
            else
            if((0-En1_L)<30)
             {
                SPEED_P=30;
                SPEED_D=20;
             }
```

```
            else
            if((0-En1_L)<40)
             {
                SPEED_P=35;
                SPEED_D=20;
             }
            else
            if((0-En1_L)<50)
             {
                SPEED_P=45;
                SPEED_D=25;
             }
            s=temp+(SPEED_P*En1_L+SPEED_D*(En1_L-En2_L))/100;//35
            if(s>0&&s<550)
              {
                  PWMDTY23=s;
              }
            else
            if(s<=0)
              {
                  PWMDTY23=20;
              }
            else
            if(s>=500)
              {
                  PWMDTY23=550;
              }
    }
else
{
             PWMDTY45=0;
             if(En1_L<10)
             {
                SPEED_P=20;
                SPEED_D=10;
             }
            else
            if(En1_L<20)
             {
                SPEED_P=25;
                SPEED_D=15;
             }
            else
            if(En1_L<30)
```

```
            {
               SPEED_P=30;
               SPEED_D=20;
            }
          else
          if(En1_L<40)
            {
               SPEED_P=35;
               SPEED_D=20;
            }
          else
          if(En1_L<50)
            {
               SPEED_P=45;
               SPEED_D=25;
            }
            s=temp+(SPEED_P*En1_L+SPEED_D*(En1_L-En2_L))/100;
            if(s>0&&s<500)
               {
                   PWMDTY23=s;
               }
            else
            if(s<=0)
               {
                   PWMDTY23=20;
               }
            else
            if(s>=500)
               {
                   PWMDTY23=500;
               }
   }
}

void speedchoice()
{
   if(straight_flag==2)
      {
         SPEED=220;
      }
   else
   if(straight_flag==1)
      {
         SPEED=190;
```

```
      }
   else
   if(straight_flag==0)
      {
         SPEED=190;
      }
   /*else
   if(fabs1(STEER,steercenter)<400)
      {
         SPEED=190;
      }
   else
   if(fabs1(STEER,steercenter)<500)
      {
         SPEED=200;
      } */
   else
      {
         SPEED=180;
      }
  //*SPEED=100;
  //PWMDTY45=SPEED;
    speedadjust();
}
void CONTROL()
{
t=5;
last_row=40,end_row=0,end_line=0;
straight_flag=3;
_left_flag=0;
_right_flag=0;
_left_count=0;
_right_count=0;
for(row=0;row<60;row++)
   {
      black_line_left[row]=0;
      black_line_right[row]=0;
      black_line_center[row]=0;
      black_line_width[row]=0;
   }
blacklineget();
//t=Straight_line(2,20);
control_flag=0;
Cross();
```

```
for(row=3;row<60;row++)
  {
    if(black_line_center[row]>5&&black_line_center[row]<75)
      {
        end_row=row;
        end_line=black_line_center[row];
      }
  }
 // if(cross_flag==1)
 // {
   //Cross_function();//last_row=30;control_row=55
  //}

for(row=3;row<last_row;row++)
  {
     if(black_line_center[row]>5&&black_line_center[row]<75)
       {
         control_line=black_line_center[row];
         control_flag++;
         control_row=row;
       }
  }
 if(fabs1(end_line,control_line)>25&&end_row>50)
  {
    control_flag=0;
    for(row=3;row<40;row++)
      {
        if(black_line_center[row]>5&&black_line_center[row]<75)
          {
            control_line=black_line_center[row];
            control_flag++;
            control_row=row;
          }
      }
  }
 if(control_flag>5&&fabs1(precontrol_line,control_line)<15)
  {
   steerprotect();
  }
 if(count>500)//straight_flag==2&&
  {
    start_line();
  }
```

```
  if(start_flag==1)
    {
       //stop();
    }
  if(stop_flag==0)
    {
       PWMDTY23=230;
       //speedchoice();
    }
  En3_L=En2_L;
  En2_L=En1_L;
  precontrol_line=control_line;
}
void main (void)
{

    DisableInterrupts;
    init_function();
    init_function1();

    gpline=0;                                   //一场中行计数清零
    gpflag=0;                                   //清除场采集结束标志
    gppoint=0;

    PACNT = 0;
    for(;;)
    {
    PACNT = 0;
    gpline=0;                                   //一场中行计数清零
    gpflag=0;                                   //清除场采集结束标志
    gppoint=0;
    while((PTH&0x01)!=0) ;                     //等待场同步信号到来
    G1=0x80;
    EnableInterrupts;                          //开中断

    TC0=TCNT+0x1500;//0x11a0;                          //等待场消隐结束
    TFLG1=0xFF;                                 //清楚计数完成标志
    TIE=0x01;                                   //开启 TC0 中断（interrupt 8）
    while(gpflag==0);
    DisableInterrupts;


    speed_now=(int)(PACNT*2.67);
    //PWMDTY45=150;
```

```
      // PWMDTY67=steerleft;
   //PWMDTY1=100; 反转
   CONTROL();
   }
}

// Vector 6 Interrupt:start get a line
interrupt 6 void line_start(void)
{
   IRQCR=0;                                    //关闭行中断
   if(gpline==70) {
      gpflag=1;                                //场结束标志位
      count++;
      if(count>65534)
         {
            count=0;
         }
   }
   else {
      gppoint=0;                               //一行中点计数清零
      TC1=TCNT+0x0015;//0x0015;          //等待行消隐结束          //越小，图像左移
      TFLG1=0xFF;                              //清除计数完成标志
      TIE=0x02;                                //开启 TC1 局部中断（interrupt 9）
   }
}

// Vector 8 Interrupt:Timer0
interrupt 8 void timer0(void)
{
   TIE=0;                                      //关闭定时器局部中断
   IRQCR=0x40;                                 //开启行同步中断（interrupt 6）

}


// Vector 9 Interrupt:Timer1
interrupt 9 void timer1(void)
{

   unsigned char *p;
   TIE=0;        //关闭定时器局部中断
   p=&(pic[gpline][0]);
   ATD0CTL2=0x40;
   ATD0CTL5=0x20;
```

```
   while(gppoint!=80)
   {
      while(!ATD0STAT2_CCF0);
      *p=ATD0DR0H;
      p++;
      gppoint++;

   }
    if(gpline <5)
   TC3=TCNT+0x0460;//39f;      //if(gpline<=49)
   else
   if(gpline< 20)
   TC3=TCNT+0x0300;//39f;
   else
   if(gpline< 25)
   TC3=TCNT+0x0200;//39f;
   if(gpline< 60)
   TC3=TCNT+0x0160;//39f;
   else
   TC3=TCNT+0x008c;

   gpline++;
   TFLG1=0xFF;            //清除计时完成标志
   TIE=0x08;              //开启 TC2 局部中断（interrupt 10）
   }

// Vector 10 Interrupt:Timer2
interrupt 11 void timer2(void)
{
   TIE=0;                 //关闭定时器局部中断
   IRQCR=0x40;            //开启行同步中断（interrupt 6）
}
```