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：　华南理工大学

队伍名称：　木棉七号

参赛队员：　陈宏论

　　黄华灿

　　杨开宇

带队教师：　陈　安

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“飞思卡尔”杯全国大学生智能汽车竞赛有关保留、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：_____

目录

第一章 绪论

1.1 前言

近年来，丰田与微软结盟欲打造下一代互联汽车平台，将多款产品整合应用于汽车；另外，谷歌开发的无人驾驶汽车已在各种路况下成功行驶，并宣称将进入大规模的测试阶段。在国内，我国自主研制的红旗 HQ3 无人车也成功进行了一系列的自主驾驶测试……这表明：汽车智能化已经大势所趋，越来越多的车企已经把它作为未来市场竞争的一个着眼点。而消费者所期待的汽车智能化除了能给驾车带来方便和舒适外，更重要的是能为安全保驾护航。

智能汽车是一个集环境感知、规划决策、多等级辅助驾驶等功能于一体的综合系统，它集中运用了计算机、现代传感、信息融合、通讯、人工智能及自动控制等技术，是典型的高新技术综合体。目前对智能车辆的研究主要致力于提高汽车的安全性、舒适性，以及提供优良的人车交互界面。近年来，智能车辆已经成为世界车辆工程领域研究的热点和汽车工业增长的新动力，很多发达国家都将其纳入到各自重点发展的智能交通系统当中。可以预见，随着汽车电子工业的快速发展，智能车将占据越来越大的汽车市场领域。

作为全球最大的汽车电子半导体器件供应商，飞思卡尔半导体公司一直致力于汽车电子半导体器件的开发与推广。飞思卡尔是嵌入式处理解决方案的全球领导者，提供业界领先的产品，不断提升汽车、消费电子、工业和网络市场。飞思卡尔在不同的半导体器件市场拥有领先的地位。其推出的 8、16、32 位微处理器依靠较高性能在全球拥有领先地位，特别是最近推出的 32 位 Kinetis K 系列 Cortex-M4 微处理器，是基于 ARM® Cortex™-M 结构的低功耗，高性能、可兼容的微控制器，非常适用于中高档智能汽车电子控制系统。

1.2 竞赛背景

“ 飞思卡尔杯 ”全国大学生智能汽车竞赛是由教育部批准并委托教育部高等学校自动化专业教学指导分委员会主办，飞思卡尔公司协办，面向全国大

学生的重要赛事。至今已成功举办了六届。它是以迅猛发展的汽车电子为背景 ，涵盖了自动控制、模式识别、传感技术、电子电气、计算机、机械等多个学科交叉的科技创意性比赛。根据比赛章程,全国大学生智能汽车竞赛是在统一汽车模型平台上，使用飞思卡尔半导体公司的 8 位、16 位、32 位微控制器作为核心控制模块，通过增加道路传感器、设计电机驱动电路、编写相应软件以及装配模型车，制作一个能够自主识别道路的模型汽车。改装后的模型汽车按照规定路线行进，以完成时间最短者为优胜。竞赛以“ 立足培养，重在参与，鼓励探索，追求卓越 ”为指导思想，旨在促进高等学校素质教育，培养大学生的综合知识运用能力、基本工程实践能力和创新意识，激发大学生从事科学研究与探索的兴趣和潜能，倡导理论联系实际、求真务实的学风和团队协作的人文精神，为优秀人才的脱颖而出创造条件。竞赛集科学性、趣味性和观赏性于一体，吸引了大批同学参与其中，受到了广泛的欢迎。

1.3 论文概述

本文将以飞思卡尔 MK60DN512 单片机作为核心控制器，以面阵 CMOS 数字摄像头作为图像传感器，获取跑道图像，运用数字图像处理技术进行图像的分割和赛道特征的提取，最后通过自适应模糊控制器控制模车转向，使车模能按照比赛规定的跑道和规则最快地完成任务。智能车系统包括图像采集、跑道图像预处理和跑道特征提取算法、自适应模糊控制器、PID 控制器、大功率电机驱动器和智能车调试监控系统等几个子系统组成。

第二章 智能车整体系统设计方案概述

大体上，模型车系统可分为图像采集、坡道检测、微控制器核心处理、转向控制、转速控制、车体实时速度采集六个部分。系统结构如下图所示：

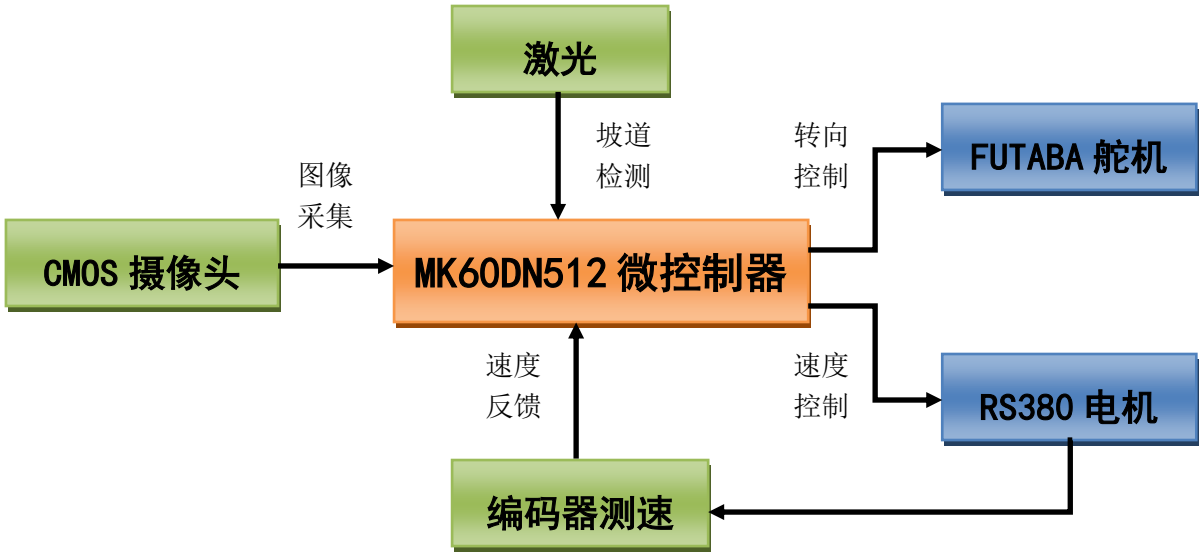


图 2 系统结构图

2.1 智能车整体布局

作为赛车，要求在特定的跑道上最快速度完成任务。这对智能车的整体组装提出比较高的要求。本届比赛摄像头组采用由东莞市博思电子数码科技有限公司提供的 G768 车模。基于此车模，总的来说，在布局上模型车应该尽量做到结构扎实、重心低、重量分布对称、差速灵敏、齿轮传动恰到其分、轴转动顺滑等等。我们经过大量拆装测试，最后确定了模型车的布局如下比较优。

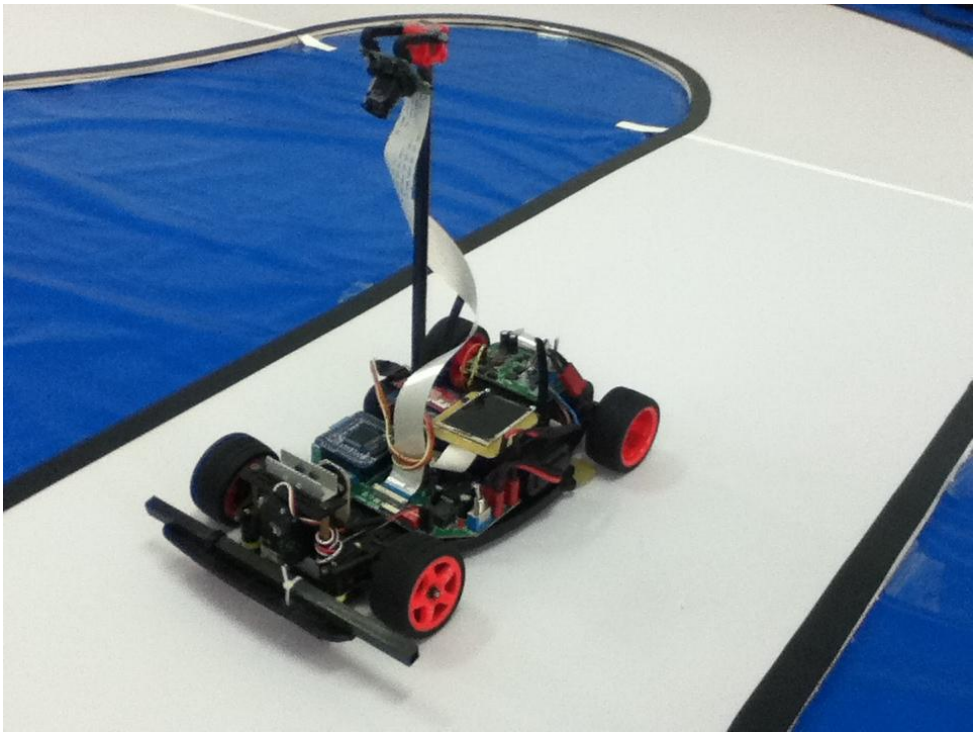


图 2-1 智能车整车布局

2.2 控制系统硬件总体设计

模型车系统实际上是摄像头传感器作为输入，舵机控制量和电机控制量作为输出，编码器输出为速度反馈的闭环控制系统。一套稳定的电路设计是该实现该系统的基础。我们最终确定的电路硬件系统如下：

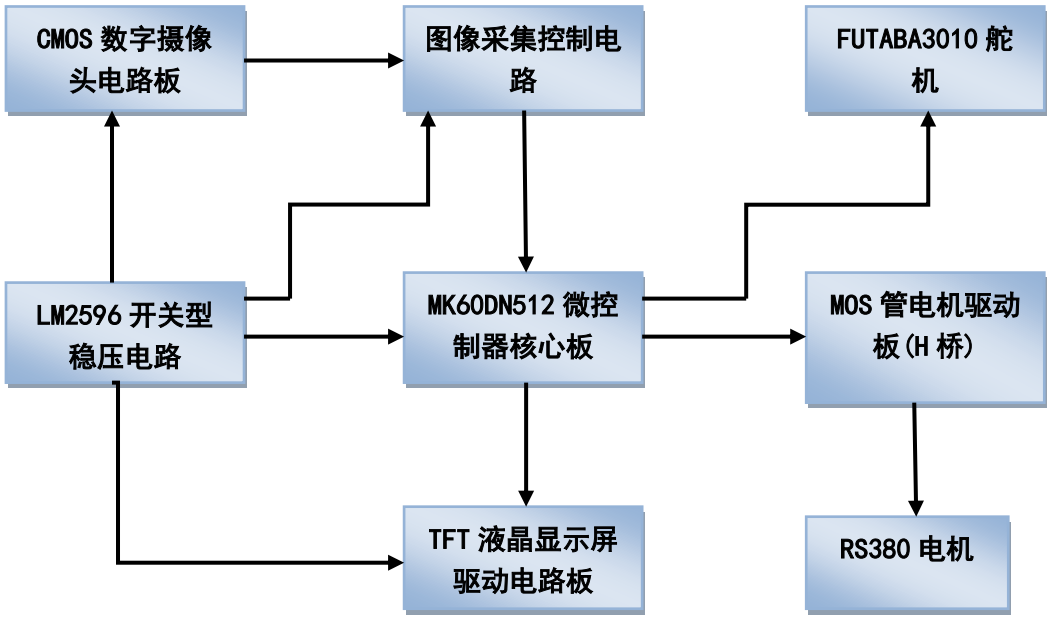


图 2-2 智能车硬件结构图

2.3 控制系统软件总体设计

车模控制软件的设计是决定比赛成绩的重中之重。智能车控制软件主要由图像采集和与处理、赛道特征提取、转向控制和车速控制程序四部分组成。车模要在比赛中取得好成绩，离不开快速的控制响应、良好的行车路线规划和准确的速度控制这三个关键点。要完成上述目标，就必须有效地融合智能车的硬件和软件，进行充分的测试、标定和实验，不断改进控制算法。我们设计的软件控制系统结构流程如下：

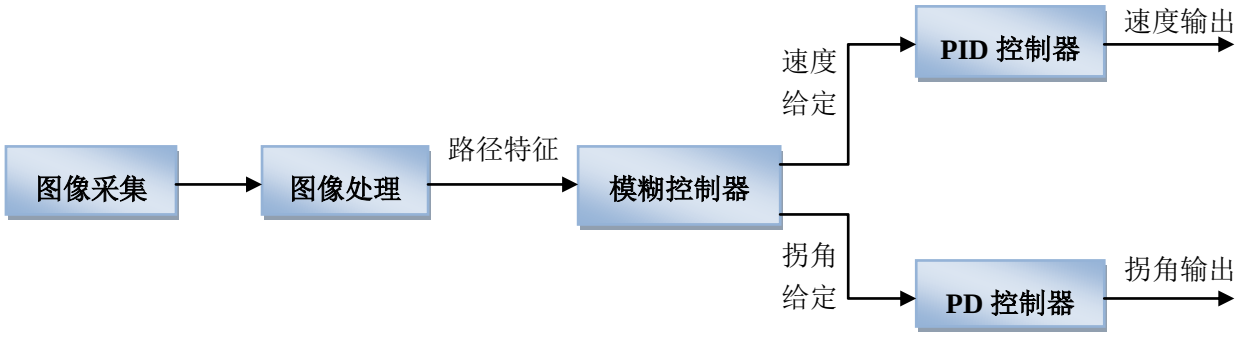


图 2-3 智能车软件结构图

第三章 模型车机械结构介绍与优化调校

3.1 车模机械结构的优化

在汽车研究中，汽车的四轮定位和悬挂的设定对行使性能和驾驶舒适性有着巨大的影响。在赛车比赛中，机械工程师会根据每场比赛跑道的特征和驾驶员的驾驶习惯对赛车的轮胎定位和悬挂进行针对性的调整。本届飞思卡尔杯智能车大赛采用竞赛组委会统一提供的 1/10 的 Matiz 仿真车模，控制结构采用前轮转向，后轮驱动方案。该车模结构比较简单，车体可调的参数较少，为了尽量挖掘该车模的最大潜力，我们对前轮定位、舵机安装和后桥连接进行了相应的改造。经测试，改造后车模的加减速和拐弯性能大大改善。

图 3-1 第七届摄像头组组委会统一提供车模

3.2 前轮定位参数的调整

该车模前轮可调的参数包括主销后倾角、主销内倾角、前轮外倾角和前束四个参数。四轮定位的作用是保持车辆可以直线行驶，帮助方向盘回正，减少悬挂的损耗等，是车辆必须的参数。

主销后倾(Caster)是指主销装在前轴，上端略向后倾斜的角。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，但过大的主销后倾角会导致转向沉重。

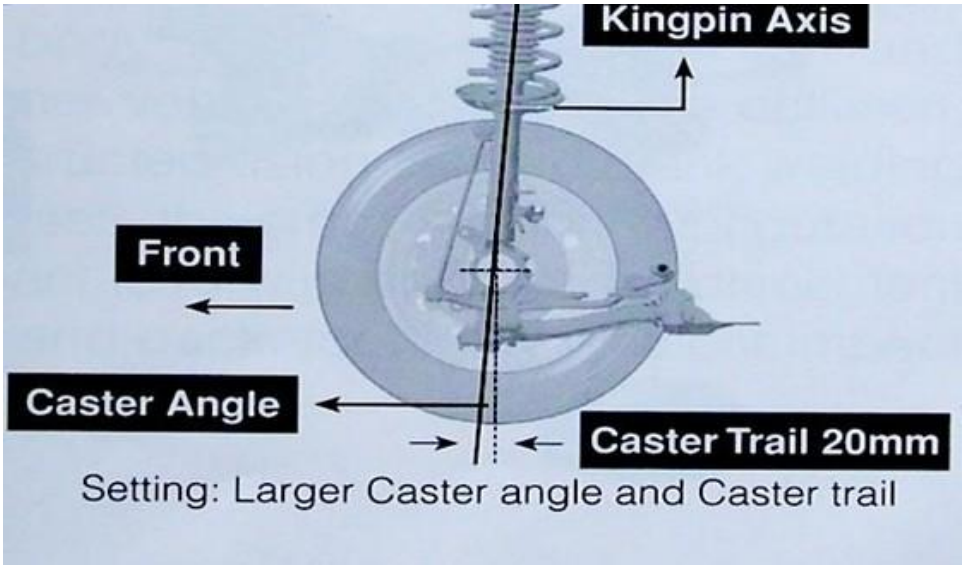


图 3-2-1 车辆主销后倾示意图

经我们测试，过小的后倾角会稍稍影响模型车高速行驶的稳定，过大的后倾角会引起拐弯滞后，实测大约 5° 左右效果最好。

主销内倾(Kingpin Inclination)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。另外，

除了这个作用之外，主销内倾还能够因为主销轴线延长线跟路面相交的点与车轮接触地面的那个点的距离缩短，从而使转向力臂增长，让转向更轻便。

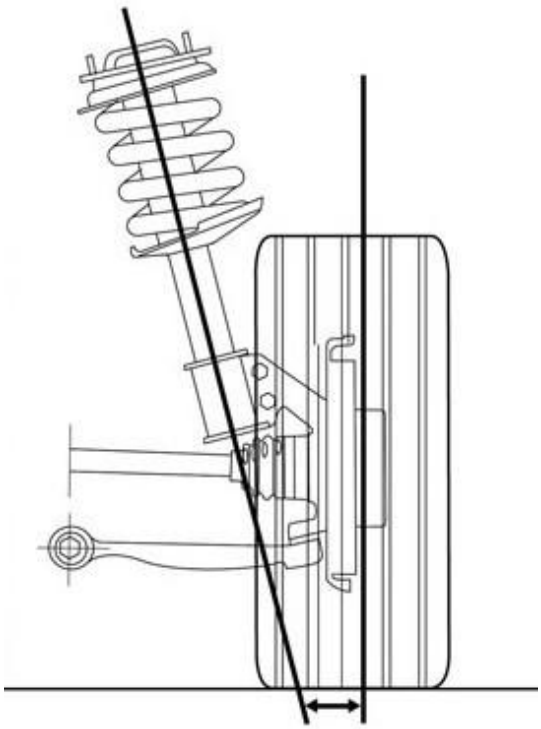


图 3-2-2 车辆主销内倾示意图

主销内倾这个角不易调，效果也难观察，我们保留了模型车内倾大约 3° 这样的角度。

前轮外倾角(Camber)对赛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。前轮外倾角俗称“外八字” 。如果车轮垂直地面一旦满载就易产生变形,可能引起车轮上部向内倾侧，导致车轮联接件损坏。所以事先将车轮校偏一个外八字角度，这个角度约在 1 ° 左右。但是，对于注重车子转弯性能的赛车来说，为了提高车辆在转向时的极限能力，一般把

前轮调成内倾。

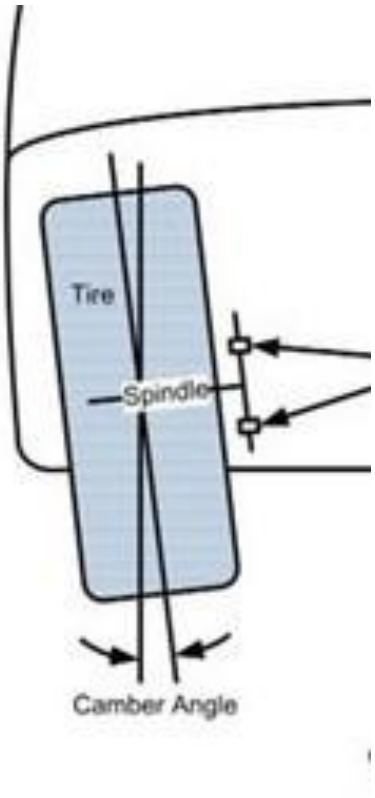


图 3-2-3 车辆前轮外（内）倾角示意图

前轮外倾角对模型车影响较大，为提高模型车转弯性能，我们最终调节其为内倾 5° 左右整体效果较优。

所谓前束(Toe)是指两轮之间的后距离数值与前距离数值之差， 也指前轮中心线与纵向中心线的夹角。前轮前束的作用是保证汽车的行驶性能，减少轮胎的磨损。前轮在滚动时，其惯性力会自然将轮胎向内偏斜，如果前束适当，轮胎滚动时的偏斜方向就会抵消，轮胎内外侧磨损的现象会减少。一般来说，前轮会被调教成正前束(Toe-In)，因为正前束可以让车轮保持直线行驶。

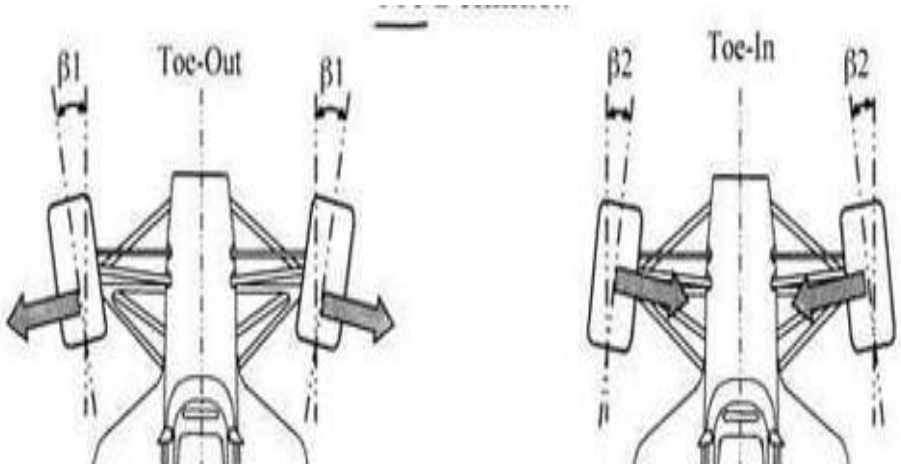


图 3-2-4 车辆前轮前束示意图

为保持模型车直线行驶的稳定性，我们把前束稍稍调节为 Toe-In，实测效果良好。

如上述，最终我们通过减小主销后倾角，加大主销内倾角，减小前轮外倾角，加大前束 Toe-In，使车模适应于在不同的路径行驶，而且车模转向性能也有了一点提高。最终效果如下图：

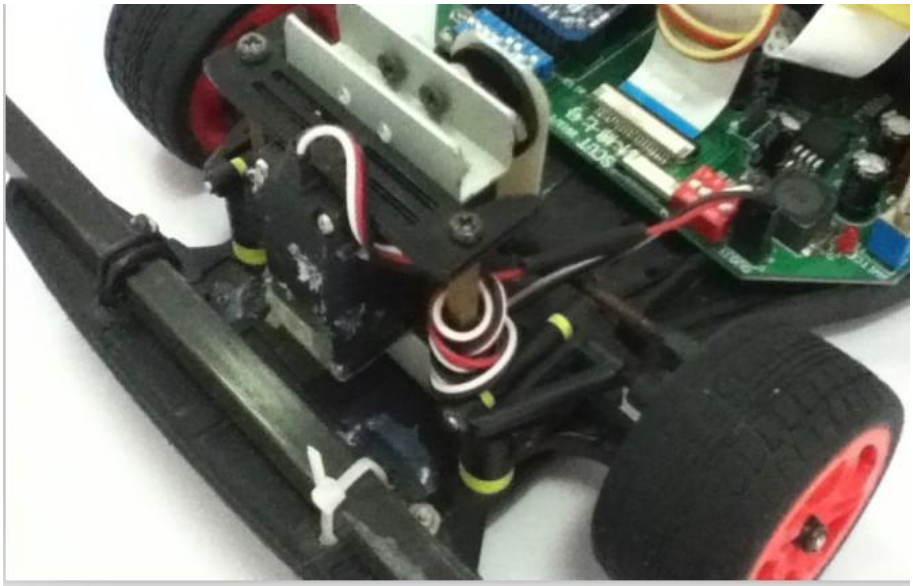


图 3-2 前轮悬挂

3.3 后桥连接调整

模型车原装的单悬挂系统如下图。该悬挂系统比较松垮，引起车模不稳定，而且它的位置妨碍了传感器和电路板的安装。于是，我们把该悬挂拆去了，后桥改用我们自己制作的 PCB 板连接件进行硬连接。这样，成功腾出位置安装电路板，而且使得车模变得更加简洁美观，另外，车模总体性能有些提高。

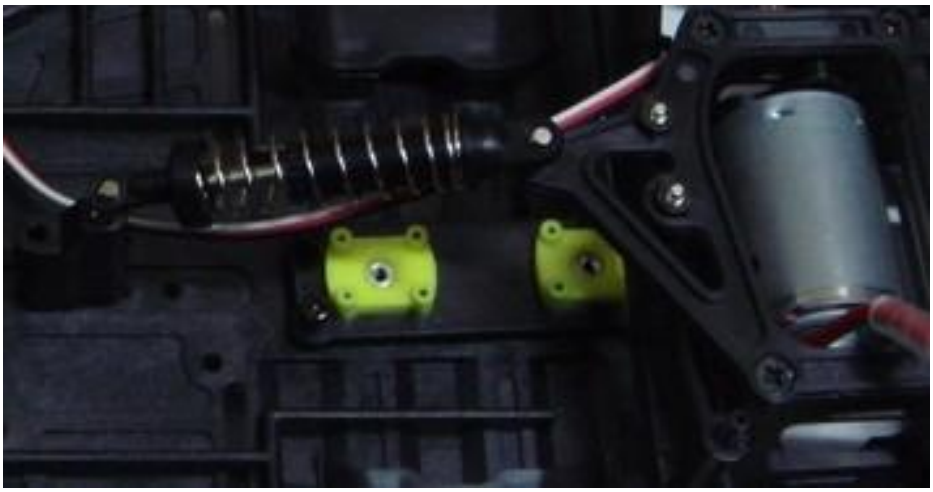


图 3-3-1 车模原装悬挂

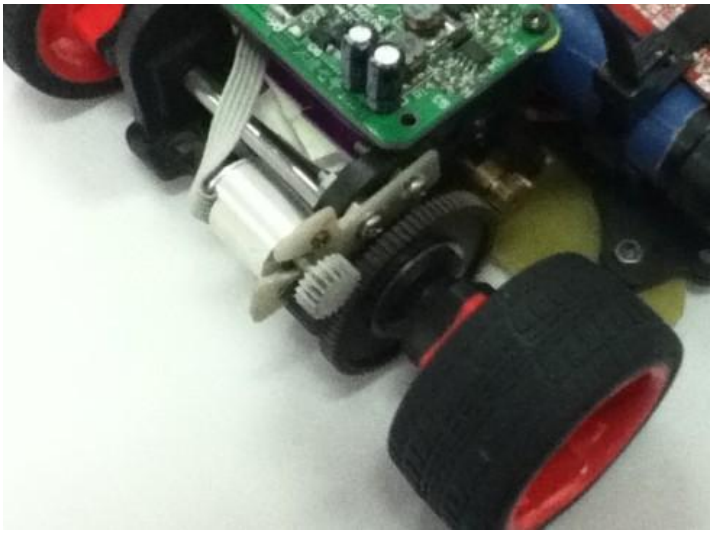


图 3-3 改造的后桥连接

3.4 舵机安装固定

为了保护舵机不在卡舵的时候损坏舵机齿轮组，原配的舵机和转向连杆之间的连接器上设有缓冲装置。但是缓冲装置有弹性会导致转向迟钝，故我们利用覆铜板自行制作了舵机的力臂，用于舵机盘和转向连杆之间的连接。此外，由于在一定的工作电压下舵机的转动速度是一定的，增长舵机力臂可提高舵机的摆动速度 。同时，由于舵机输出的扭力一定，力臂太长会影响转向的力度，使舵机“变软” ，我们制作了长度适中的舵机力臂，并把舵机安装在车头正中的位置，使左右转向对称。

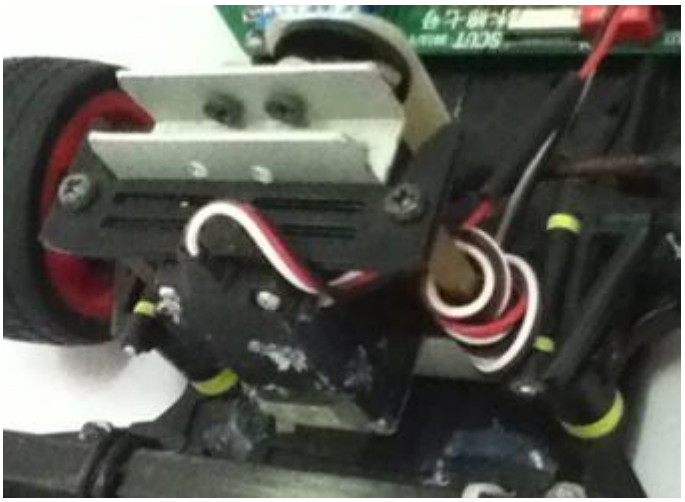


图 3-4 舵机安装图

3.5 摄像头安装固定

我们自行采用高强度轻量化的材料制作摄像头的支撑机构，降低高位的重量，同时安装位置适当调整了车模前后的重量分布，以获得更好的行驶性能。

图 3-5 摄像头安装图

3.6 差速器的调节

比赛使用的模型车设置有后轮差速机构，其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。本模型车后轮轴上配备的是简易的差速器，且比赛规则规定不能更换，所以在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，从而

大大影响车模的寻迹性能。过松会使差速器严重打滑，损失电机输出的驱动扭矩，过紧又会损失差速能力。一个性能优良的差速器应该具备不打滑、不卡死、差速反应灵敏等特点。

图 3-6 差速器

3.7 电路板安装，车体重心调节

为了均衡模型车重量分配，我们的电路板安装在车模底板上靠前的位置，这样可以保证四轮合适的抓地力。

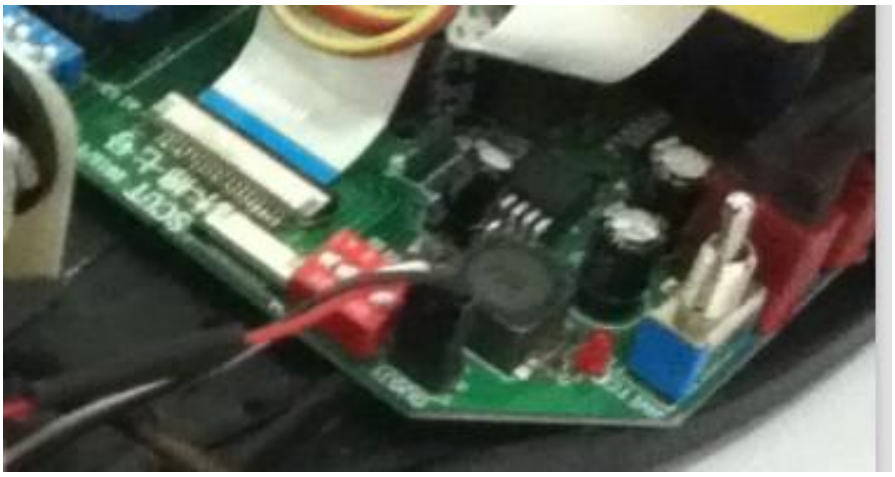


图 3-7 电路主板安装图

第四章 硬件系统设计

智能车硬件系统由稳压电路、K60 核心板、摄像头供电、编码器供电以及拨码盘和液晶显示器等部分组成。智能车的唯一供电电源由组委会规定的 7.2V 2000mAh 的专用镍镉电池提供。

4.1 稳压电路设计

4.1.1 稳压电路方案论证

常用的电源有串联型线性稳压电源（LM2940、7805 等）和开关型稳压电源（LM2596、LM2576 等）两大类。前者具有纹波小、电路结构简单的优点，但是效率较低，功耗大；后者功耗小，效率高，但电路却比较复杂，电路的纹波大。

备赛初期我们采用 LM2940 作为主电路 5V 电路的稳压芯片，电路非常简单，但是发现 LM2940 发热比较严重，效率较低。本着探索的精神，我们后来尝试了效率明显高些的 LM2596 并搭设了个开关型稳压电路。由于全备的滤波电路和精心的 PCB 布线，我们测试发现电路纹波在可接受范围，而且实际使用也正常稳定。重要的是，LM2596 工作时的发热极小，避免了线性稳压的发热问题，同时，其最大 3A 电流的输出也存在着明显优势。于是我们最终采用了 LM2596-5V、LM2596-3.3V 两个稳压芯片搭建的稳压电路，分别提供 5V 和 3.3V 电源。

4.1.2 稳压电路设计

以下稳压电路的输入为镍镉电池 7.2V，输出一路 5V 电源和一路 3.3V 电源，分别为摄像头、编码器、单片机核心板和其他外围电路供电。

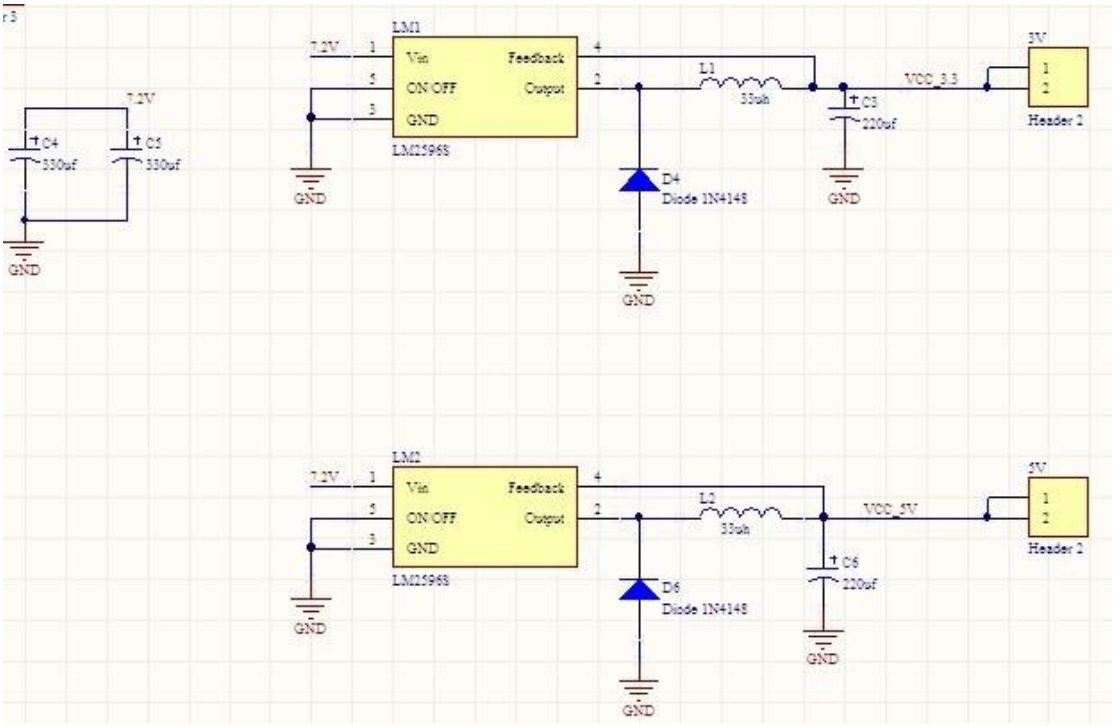


图 4-1 稳压电路原理图

4.2 摄像头

CCD 与 CMOS 传感器是被普遍采用的两种图像传感器，两者都是利用感光二极管进行光电转换，将图像转换为数字数据，而其主要差异是数字数据传送的方式不同。CCD 摄像头在灵敏度、分辨率、噪声控制等方面优于 CMOS 传感器，但工作电压为 12V，需要对电池电压进行升压处理后再给它供电，增加了电源管理电路的复杂程度。而 CMOS 摄像头拥有集成度高，功耗低，图像稳定性较高等优势。CMOS 摄像头可分为数字和模拟两种。相比于模拟 CMOS 摄像头输出一路模拟信号，数字 CMOS 摄像头则可以直接输出数字图像信号，使得微处理器可以直接采用一组 IO 口读取传感器发回来的图像信息。这大大简化了主板硬件电路的设计。于是我们采用了 ov7620 数字摄像头，能够达到 30 帧/S 的帧速率，提供曝光值和增益等可调参数，使得摄像头可以适应于更广泛的场合，增加了应用的灵活性。

ov7620 是 Omnivision 公司设计的一款彩色 CMOS 型图像采集集成芯片，提供高性能的单一小体积封装。1/3 英寸数字式 CMOS 图像传感器 OV7620,总有效像素单元为 664(水平方向)×492(垂直方向)像素;内置 10 位双通道 A/D 转换器,输出 8 位图像数据;具有自动增益和自动白平衡控制,能进行亮度、对比度、饱和度、γ 校正等多种调节功能;其视频时序产生电路可产生行同步、场同步、混合视频同步等多种同步信号和像素时钟等多种时序信号;5V 电源供电,工作时功耗<120mW,待机时功耗<10μW。它支持连续和隔行两种扫描方式，VGA 与 QVGA 两种图像格式;最高像素为 664×492,帧速率为 30fps;数据格式包括 YUV、YCrCb、RGB 三种,能够满足一般图像采集系统的要求。可应用于数码相机、电脑摄像头、可视电话、第三代网络摄像机、手机、智能型安全系统、汽车倒车雷达、玩具，以及工业、医疗等多种用途。

Array Elements	664 x 492
Pixel Size	7.6 x 7.6 um
Image Area	4.86 x 3.64mm
Electronic Exposure	500 : 1
Scan Mode	progressive interlace
Gamma Correction	128 Curve Settings See specifics
Minimum Illumination	2.5 lux @ f1.4 0.5 lux @ f1.4 (3000K)
S/N Ratio	> 48dB
Power Supply	5VDC, ±5%
Power Requirements	<120mW Active <10uW Standby
Package	48-pin LCC

图 4-2　ov7620 传感器

4.3 微处理器的确定

组委会规定车模控制电路必须采用飞思卡尔半导体公司的 8 位、16 位、32 位 MCU 作为唯一的微控制器。飞思卡尔不同系列的微控制器包括，32 位 Kinetis 系列；32 位 ColdFire 系列；32 位 MPC56xx 系列；8 位微控制器系列（可使用 2 片）；16 位 DSC 系列；16 位微控制器 9S12XS 系列；16 位微控制器 9S12G 系列。这样使得我们的选择大大宽泛了，可以根据自己的设计方案择优采用不同的主控芯片。我们的选择是属于 32 位 Kinetis 系列的 K60DN512。根据我们的测试和市场反应，尽管飞思卡尔的其他系列微控制器拥有较高的性能，同时由于推出得比较早，某些系列的使用者较多，应用资料也较为丰富，基于 ARM Cortex-M4 内核的 Kinetis 系列依靠强劲的内核和丰富的外围设备仍有些优势。

K60DN512 芯片是 Kinetis 系列 K60 子系列的典型芯片， 是 Kineits 系列中集成度最高的芯片。K60DN512 具有丰富的通讯接口、AD 转换电路和外围控制电路。

其特性如下：

（1）ARM Cortex-M4 内核，主频高达 100MHZ。

（2）32 路 DMA 供外设和存储器使用，大大提高 CPU 利用率；

（3）10 种低功耗模式，包括运行，等待，停止和断电；

（4）512K Flash 和 128K SRAM；

（5）集成硬件和软件看门狗，硬件加密电路和 CRC 电路；

（6）33 路单路和 4 路差分的 16 位 AD 转换，2 路 12 位 DA 转换；

（7）8 路电机控制，2 路方波解码，4 路可编程定时器；

（8）SD 卡主机控制器，6 路 UART，IIC，IIS，SPI，CAN；

（9）USB2.0 全速和高速接口，支持 OTG；

（10）IEEE1588 以太网接口，支持 MII 和 RMII 通讯；

（11）工作电压 1.71V~3.6V，工作温度-40°~105°；

（12）多达 100 路 GPIO 引脚，所有 GPIO 引脚兼容 5V 电平。

从上面的特性可以看出，K60 的性能相当的强悍，可以大大方便智能车控制系统方案的设计。

4.4 速度传感器

为了使车模以最快的时间完成比赛，这就要求车模能以高速平稳地通过直道，并以恰当的速度平稳地转弯，也就是要求车模要有较好的调速性能。采用开环调节时电机特性比较软，转速受到供电电压、负载变化等因素变化影响较大，不利于控制系统的判断，所以需要设置速度传感器，对车速进行实时检测并反馈到主控芯片以实现闭环控制。众所周知，闭环控制可以大大使电机特性变硬。检测车速的办法有很多种，例如测速电机、反射式光电检测、对射式光栅检测、霍尔测速、旋转编码器等等。我们曾使用过测速电机，测速发电机有着电路简单可靠、反应灵敏、便宜等优点。缺点是精度不够，线性度不够好等。随着调试的推进，车模对速度的准确性要求也越来越高。为提高整个控制系统的性能，我们最终采用工业用的旋转编码器进行测速。实测可知该编码器测速比测速发电机精度要高些。

图 4-4　测速编码器安装图

4.5 参数调整模块设计

智能车一旦提交到比赛组织方后便不可以对微处理器的程序作任何修改。因此，作为赛车，为了应对不同的比赛场地和不同的比赛场景，设计一个人机交互模块进行模型车的控制系统的主要参数调节显得非常重要。我们采用若干个拨码按键和按键对控制系统几个主要参数进行调节。

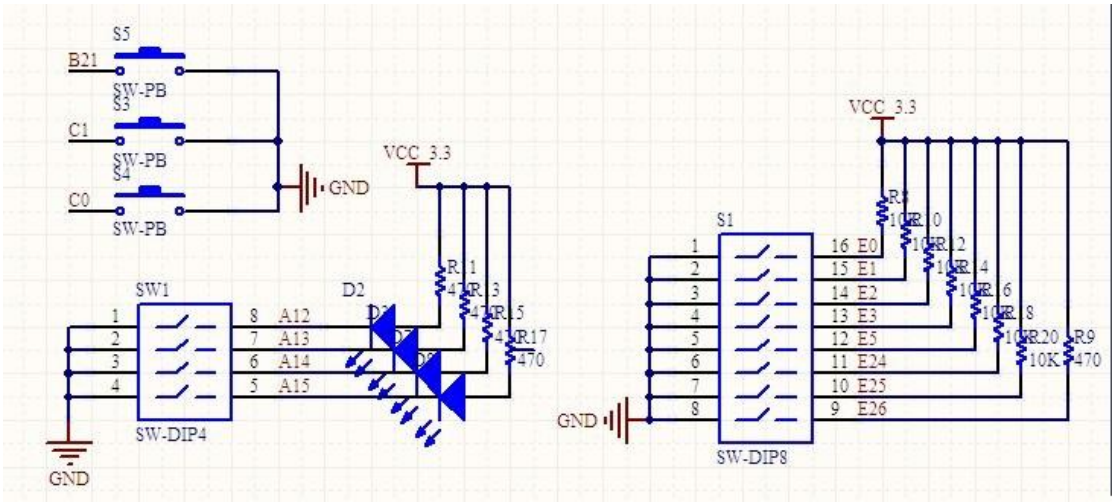


图 4-5 参数调整模块

4.6 车模状态显示屏设计

由于摄像头组模型车的特殊性，为了赛车比赛前准备调试时观察赛车的传感器等器件的状态，我们自行制作了一个 TFT 液晶显示屏并安装在模型车上。该屏的分辨率可达 240 X 320。这样，微处理器可以通过它向发车手“报告”各方面的状况。

图 4-6-1　TFT 显示屏实物

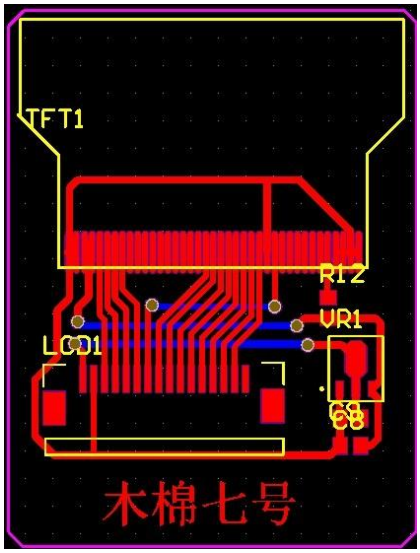


图 4-6-2 TFT 显示屏 pcb 图

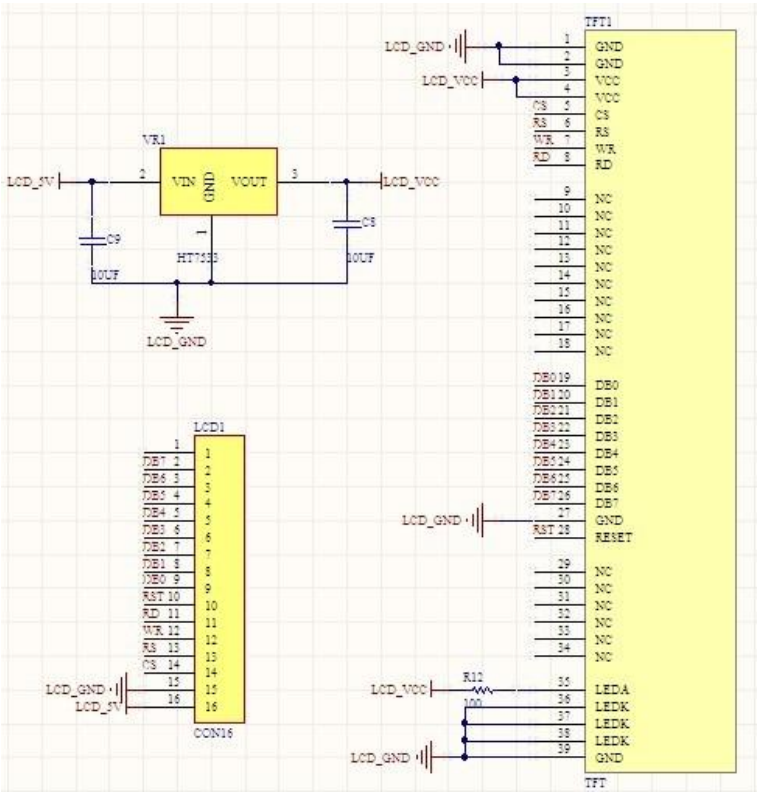


图 4-6-3　TFT 显示屏模块电路

第五章 软件系统设计

5.1 图像信号处理

5.1.1 图像采集

由于我们采用的是数字 CMOS 摄像头，为了使摄像头采集的图像信息以最快的速度、最简单的路径输送到单片机的内部 RAM 中，我们注意到了 K60 微处理器内置的 DMA 控制器。所谓 DMA 就是直接内存存取（Direct Memory Access）技术。与传统 CPU 采取轮询方式、中断方式读取外部数据不同，DMA 是通过 DMA 控制器接管数据和地址总线，根据事先设定好的源地址和目的地址，以及传送的字节数，将数据自动传送到指定的位置。数据传输完全不需要 CPU 的介入，从而大大减轻了 CPU 的压力，提高了处理器的工作效率。

摄像头数据输出时序如下，我们利用像素点时钟信号(PCLK)、行信号(HREF)、场信号(VSYNC)保证数据同步。

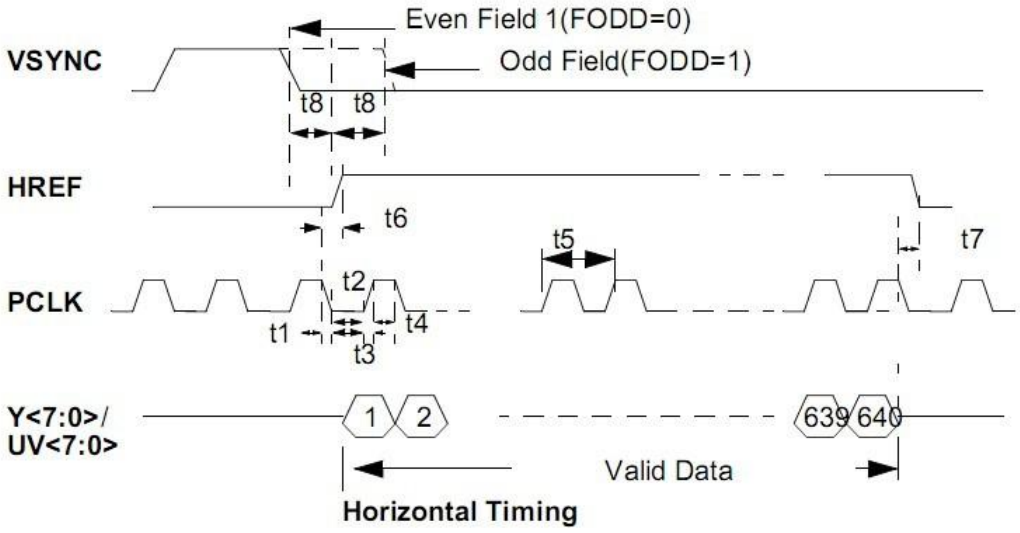


图 5-1-1 摄像头数据时序图

图像采集流程如下：

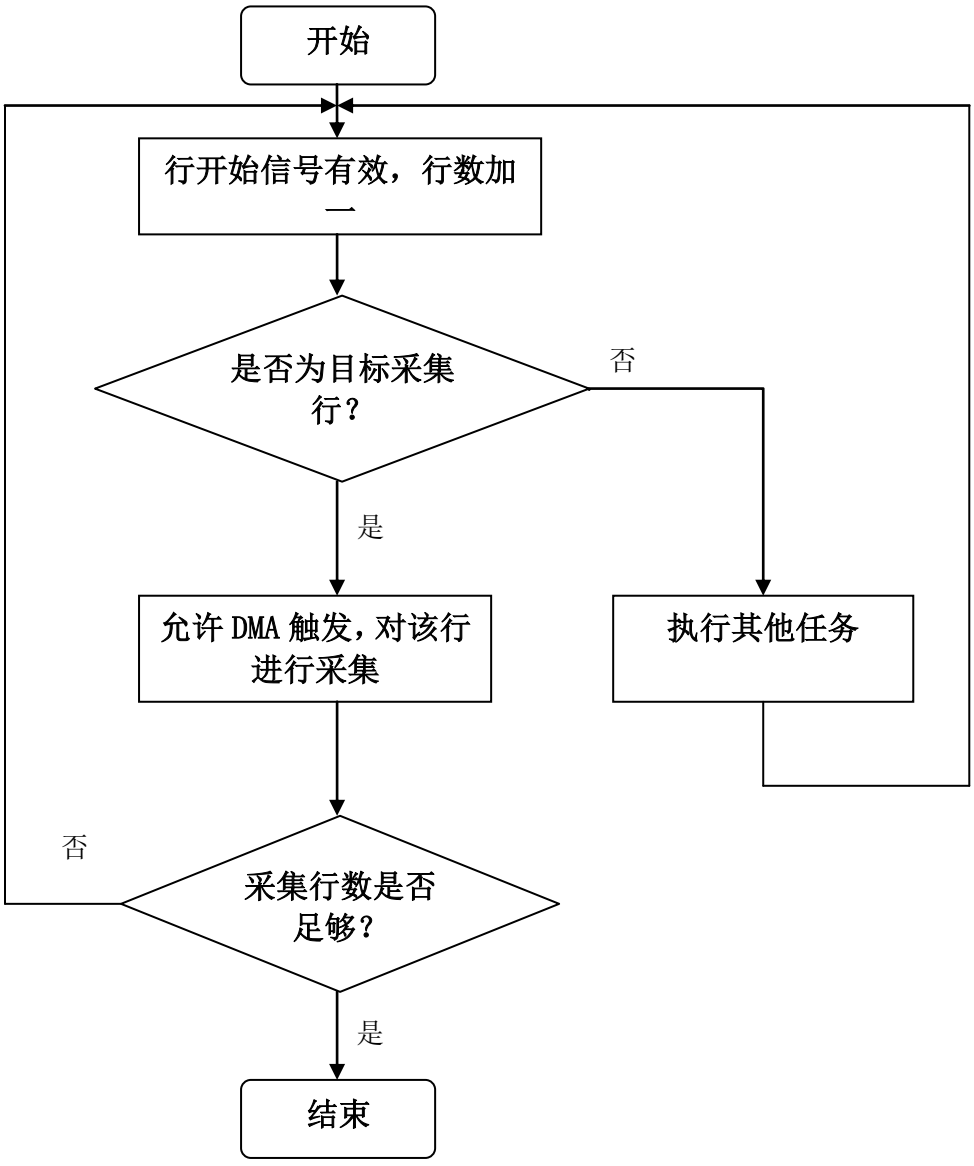


图 5-1-2　图像采集流程图

5.1.2 图像二值化阈值的获取

数字摄像头输出为一副 0~255 灰度值图，赛车需要在视场中查找出贴有黑色边缘的白色 KT 板作为前进路径。于是，为了对图像的进一步处理，我们要对采集回来的图像进行二值化。二值化是跑道检测的第一步，也是很重要的环节，在图象处理中是图象分割的范畴，跑道两边黑线能否准确的从背景中提取出来直接影响到后续样本点的采样。数字图象处理的目的之一是图象识别，而图象分割与测量是图象识别工作的基础。图象分割是将图象分成一些有意义的区域，然后对这些区域进行描述，相当于提取出目标区域图象的特征，判断图象中是否有感兴趣的目标。图象分割的基础是像素间的相似性和跳跃性。所谓“相似性”是指某个区域像素具有某些相似性，如灰度一样，纹理相同；所谓“跳变性”是指特性不连续，如灰度指突变等。从总体上说，图象分割就是把图象分割成若干有意义的区域的处理技术。

图像二值化为下面的路径提取做准备。因此二值化阈值的正确提取至关重要。鉴于跑道的设计为由黑白两种极端灰度值组成，同时，通过调整摄像头的增益和曝光值等成像参数可以使得摄像头拍摄的图像灰度值分布分离比较明显，另外为了尽量减少计算以节省算法执行时间，我们采用改进的“双峰法”获取二值化阈值。传统双峰法的原理很简单：它认为图像由前景和背景组成，在灰度直方图上，前后二景都形成高峰，在双峰之间的最低谷处就是图像的阈值所在。我们对其稍稍修改，根据图像灰度分布情况以及远近图像的灰度动态改变阈值的偏向，使得其更好地适用于不同灰度分布的场景。

5.1.3 路径的提取

5.1.3.1 路径边缘提取

智能车赛跑的跑道示意图如下：

图 5-1-3-1-1 赛道示意图

可知，准确提取跑道两边的黑线对模型车的稳定运行至关重要。考虑算法的执行效率，我们采用边缘检测方法对黑线进行提取，效果良好。所谓边缘检测，就是利用上面已经获取好的阈值对图像进行扫描，扫描到由白到黑跳变视为查找到目标。由白到黑的跳变可以对白点和黑点的数量作限制。搜线流程如下：首先，我们对图像底下 20 行按照从下到上，从中间到两边的顺序查找黑线出现的起点。若同时连续三行都在附近查到边缘，视为找到起点。然后，采用跟踪算法从下到上依次以上一行找到的黑线作为中点开窗口查找下一行的黑线，逐行遍历黑线。跟踪算法不仅可快速查到黑线，使得算法效率变高，也有一定的抗干扰能力。具体流程如下：

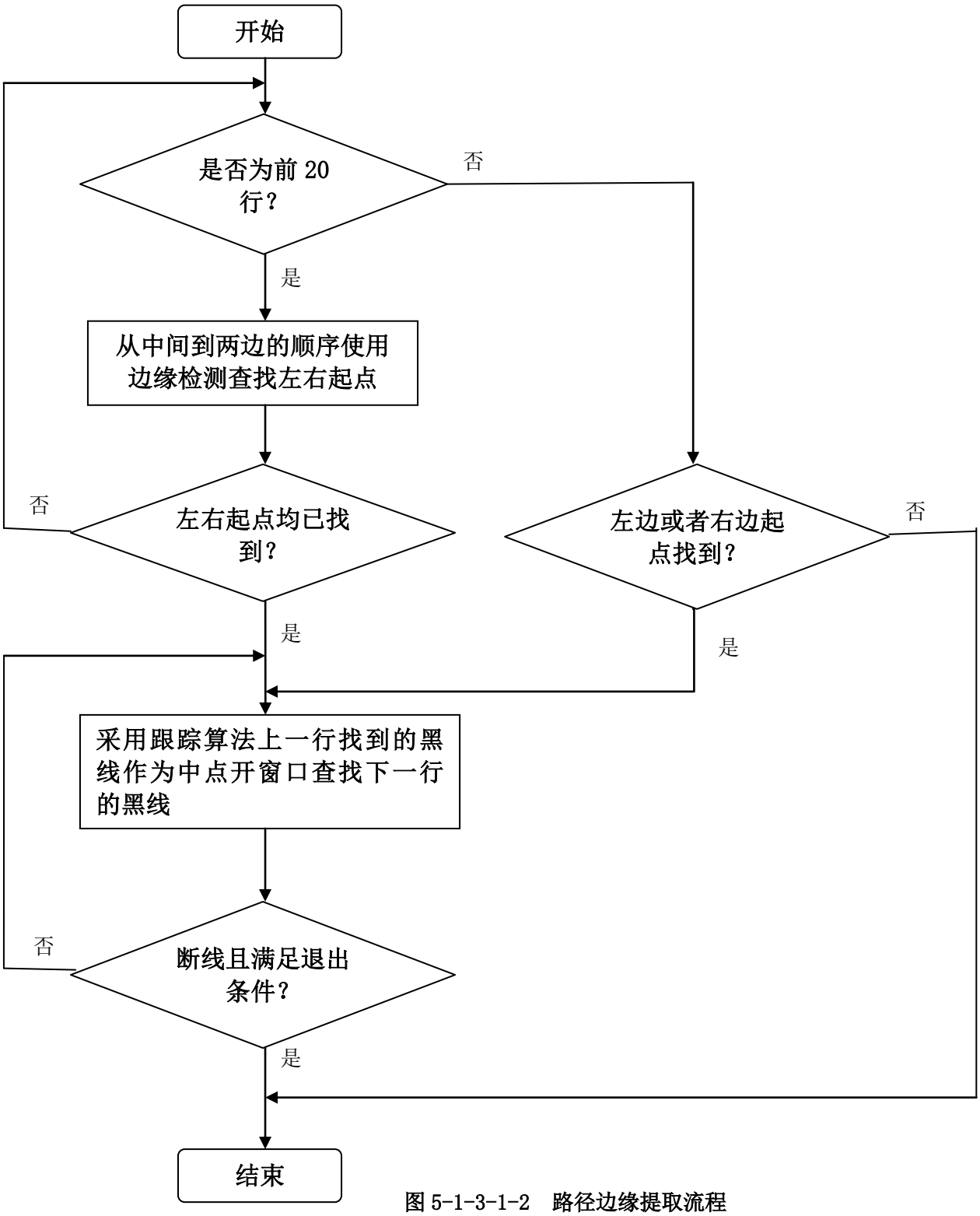


图 5-1-3-1-2　路径边缘提取流程

5.1.3.2 十字路口的处理

按照规则，智能车的赛道会出现以下的十字交叉：

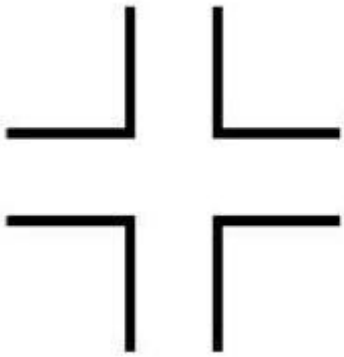


图 5-1-3-2　十字路口

当智能车行驶到十字路口时，按照上述的遍历边缘检测方法查找两边黑线会遇到黑线断续甚至错查找到错误入口的情况。这时，必须对十字路口进行识别，进行特殊处理。对于第一种断线的情况，可以轻易根据下端的短线的趋向查验是否前方是否为白色跑道知道这是十字路口；而对于第二种找错线的情况，可以通过一次差分计算，若是十字路口，将会出现一个强烈的突变点，这时可以综合另一边线的情况（遇到白色跑道断线）即可判断为十字路口。此外，对于斜入十字路口只有单线的情况，将会出现单线拥有强烈的突变点的情况，这时图像上方将可以看到十字路口入口的单边或者双边，同样可以将其识别出来。

接着，为了模型车可以平滑顺利地通过十字路口，我们采取主动补线的方法。具体是在识别到十字路口后，根据已确定的有效线段的趋向，向上搜索十字路口的入口，查找到后，再将两边黑线连接起来。这样，车模不仅可以正确通过十字路口，而且比较平滑。

5.1.3.3 终点线的识别

按照规则，智能车运行过以下的终点线应该停车：

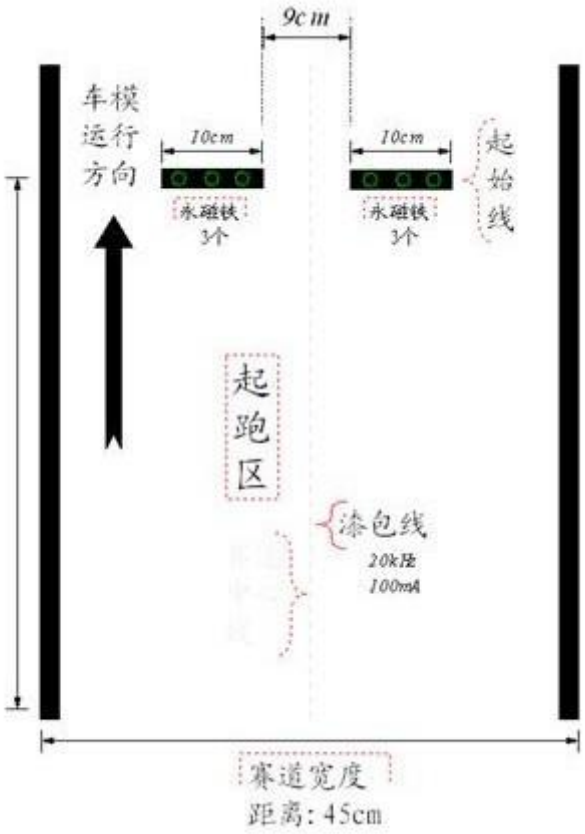


图 5-1-3-3　终点线示意图

分析可知，这是赛道上唯一会在两条黑色边缘之间会出现黑块的地方。我们使用摄像头拍摄到的图像进行终点线的识别。具体就是从左边缘向右搜索一定范围内（15~20 个点）看是否会出现黑块，同时，从右边缘向左搜索，也是一定范围内（15~20 个点）看是否会出现黑块。若有黑块，再检查其分布是否满足规则中的终点线规律，以避免干扰。注意的是，由于赛车运行速度较快，为了可靠地识别终点线，我们可以在图像底下三十行都进行这样子的搜索。这样子可以保证摄像头可以多次拍摄到起跑线，增加了可检测次数。需要注意的是，

由于使用 CMOS 数字摄像头，我们可以根据现场光线等合理调整摄像头的曝光值和增益等参数，使得摄像头可以拍摄到终点线。

5.1.4 图像透视变形的矫正

由于摄像头安装是斜视着地面，与地面垂直线有一定的角度。 因此，根据摄像头的光学成像模型，采集回来的图像将会出现梯形失真，即所谓的 “透视变形” 。

在路经的识别中，图像的透视变形会带来一系列问题：垂直线被拍摄成斜线，远处的弯道由于透视原因远处弯道被缩小导致曲率计算错误，因远近处图像比例不同导致斜率计算错误和入弯距离计算错误。因此，进行图像的透视校正对提高路径特征获取的精度十分重要。

根据几何数学建模，摄像头获取图像的成像坐标与景物实际的世界坐标有如下关系：

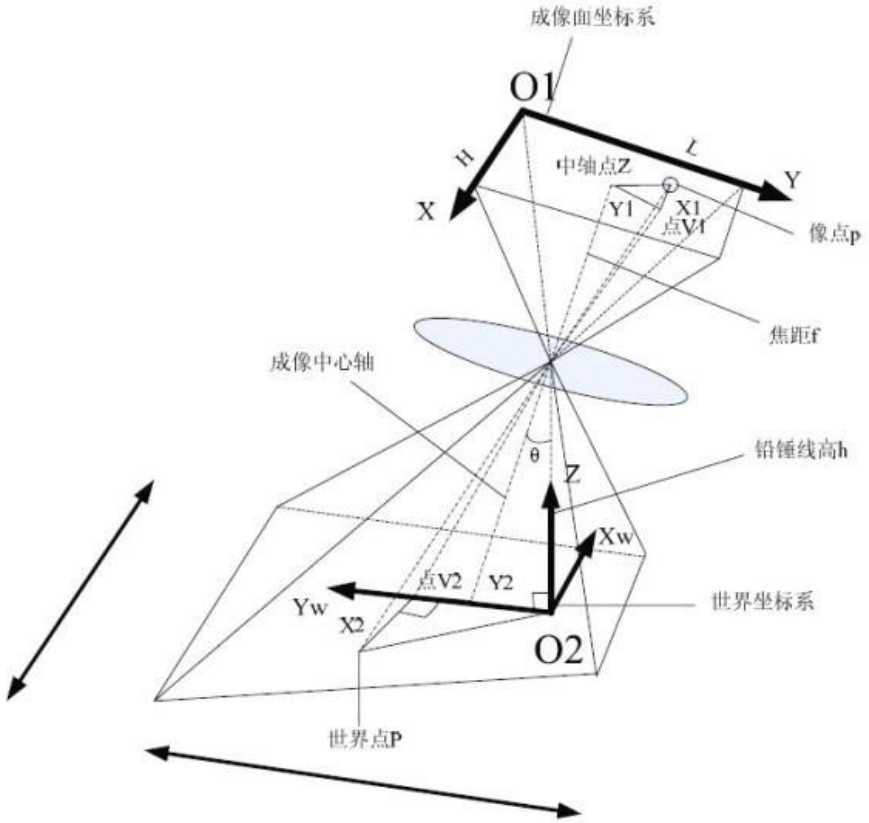


图 5-1-4　投射坐标系与世界坐标系关系

由上图可得出以下关系式：

$$\begin{cases} Y2 = \dfrac{\dfrac{c}{\tan\theta} row + a}{b - c * row} \\ X2 = \dfrac{\left(\dfrac{l}{l\_\max} line - 0.5\right)}{b - c * row} \dfrac{h}{\cos\theta} \end{cases}$$

$$b = \frac{f}{H} - 0.5\frac{L}{H}\tan\theta \qquad c = \frac{L}{H}\frac{\tan\theta}{r\_\max} \qquad a = \frac{f}{H}\tan\theta - 0.5\frac{L}{H}$$

于是，我们根据摄像头安装固定的 a,b,c, θ，可以建立图像坐标与实际坐标的对应关系，即所谓的图像“倒梯形校正”，这样可以使得更精准地获取跑道的实际走向，算出更精准的路径特征。

5.1.5 路径特征的获取

图像处理的目标是图像识别。我们得出路径的边缘信息后，将它们合成一条虚拟的路径中心线以供模型车运行。正常来说，模型车应该位于图像成像的中心轴上，通过路径中心线与图像中心轴的偏移可以计算模型车相对跑道中线的位置。另外，通过对中心线的斜率进行计算可以获得前方跑道相对车体的趋向。根据提取出的赛道这两个特征就可以合理控制模型车的转向和速度。

5.2 模糊控制器的设计与实现

5.2.1 模糊控制器的特点

经典控制论（如经典 PID 算法）可以有效地解决线性定常系统的控制问题，但是在现实世界中，有相当大数量的控制系统（如动态非线性过程），由于被控过程的数学模型难以用传统的方法准确把握，经典控制论常常难以实现较高的控制效果。由于复杂系统的变量太多且很多关系难以清晰提取，对于传统控制论来说，要求提取被控对象的精准的系统动态信息有些无能为力。

先进控制技术模糊控制利用模糊集合理论将专家知识或操作人员经验形成的语言规则直接转化为自动控制策略。作为智能控制算法中一种，模糊控制不要求精确的数学模型，人机对话能力较强，能够方便地将专家的经验与思考加入到知识模型中。简化系统设计的复杂性，特别适用于非线性、时变、模型不完全的系统上。其特点有：

1、利用控制法则来描述系统变量间的关系。

2、不用数值而用语言式的模糊变量来描述系统，模糊控制器不必对被控制对象建立完整的数学模式。

3、模糊控制器是一语言控制器，使得操作人员易于使用自然语言自然语言进行人机对话。

4、模糊控制器是一种容易控制、掌握的较理想的非线性控制器，具有较佳的适应性及强健性、较佳的容错性。

高速运动的智能车实际上属于高实时要求的非线性系统，若运用经典控制论很难到达理想的动态性能和高度的实时性。而模糊控制技术则十分适合智能车这种时变的、非线性的、高实时响应要求的系统。

5.2.2 模糊控制器的实现

模糊控制的核心是模糊控制器。它主要有以下四部分组成：

1、模糊化：这部分的作用是将输入的精确量转换成模糊化量。

2、知识库：知识库中包含了具体应用领域中的知识和要求的控制目标，即存放控制规则。

3、模糊推理：它具有模拟人的基于模糊概念的推理能力。

4、清晰化（去模糊化）：主要是将模糊推理得到的控制量（模糊量）变换为实际用于控制的精确量。

模糊控制器结构：

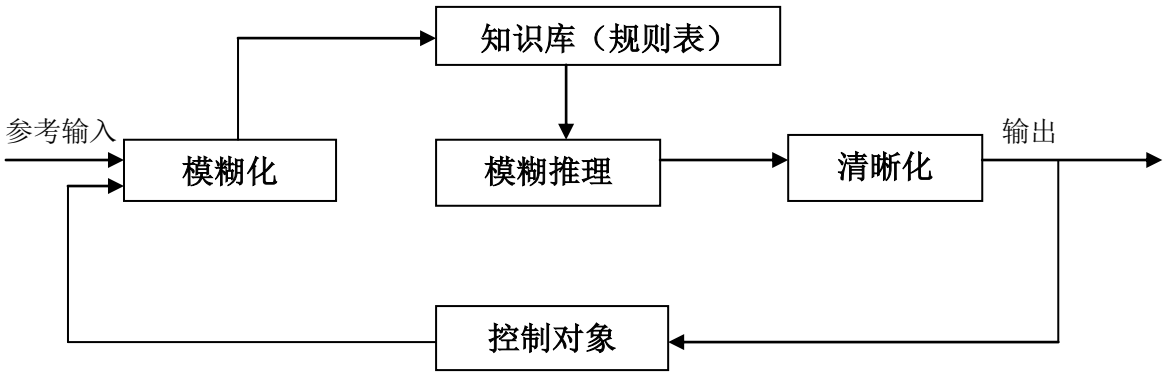


图 5-2-2 模糊控制器结构图

5.2.3 双输入双输出模糊控制器在智能车控制中的应用

由前面的图像处理可以获得模型车实时的位置偏差和角度偏差，我们把它们作为模糊控制器的输入，通过与车模性能匹配的模糊规则进行模糊推理，最后输出合理的转向给定和车速给定。流程如下：

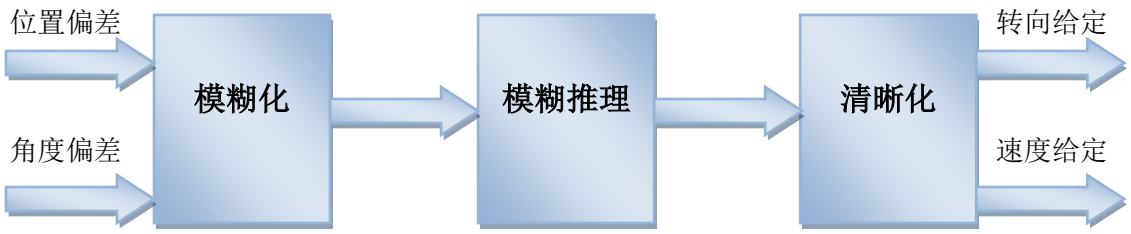
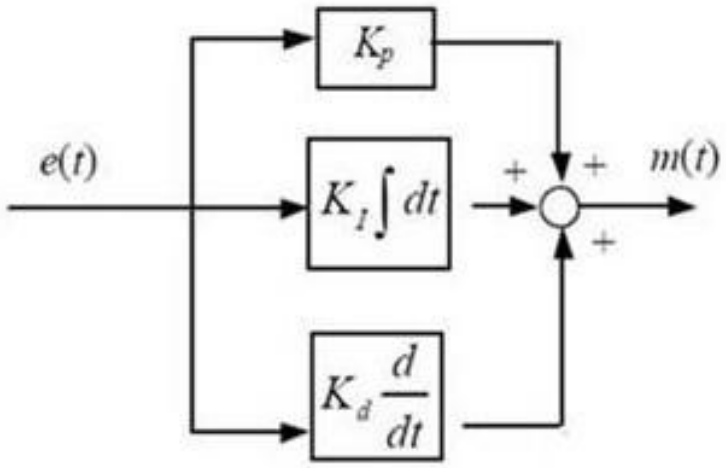


图 5-2-3 模糊控制器在智能车控制的应用

5.3 速度 PID 控制器的实现与调节

对于竞速的模型车，要实现高质量的控制效果，必须设法使车模运行速度快速跟随控制器的速度给定。经典的 PID 算法是建立在经典控制理论上的一种控制策略。它具有简单易懂，不要精确的系统模型等特点，应用广泛。PID 控制器结构如下：



速度偏差 e(t)在 PID 调节作用下，进行比例、积分、微分运算后，得出一个控制输出量 m(t)控制电机速度。

PID 控制器的数学描述为：

$$m(t)=[Kp\ e(t)+Ki\int e(t)\,dt+Kd\frac{de(t)}{dt}]$$

其中，Kp 为比例系数，Ki 为积分常数，Kd 为微分常数。

比例系数 Kp 的值增大时，系统的响应速度加快，闭环系统响应的幅值增加。Kp 值过小，系统响应明显滞后；Kp 值过大时，系统将出现震荡，不稳定。

积分常数 Ki 的值减小，系统超调量减小，而系统的响应速度将变慢。积分

环节的主要作用是消除系统的稳态误差。积分常数要根据实际合理调节。

微分常数 Kd 增大，系统的响应速度增加，同时响应的幅度也增加。因此，微分环节的主要作用是提高系统的响应速度，使系统具有一定的“前瞻性”。但要注意微分环节产生的控制量只与偏差的变化率有关，对于信号无变化，或者变化缓慢的过程则不起作用。微分常数一般比较小比较合适。

第六章 调试系统设计

对于飞思卡尔智能车比赛，对赛车程序进行编写与调试是必不可少的一个环节。好的调试软件和调试方法可以使得程序开发效率更高，更容易地解决问题。

6.1 软件开发调试系统 Keil μVision4

Keil 公司是一家业界领先的微控制器软件开发工具的独立供应商，在 2007 年被 ARM 公司收购。ARM 推出了针对各种嵌入式处理器的软件开发工具 RealView MDK 开发环境。RealView MDK 集成了业内领先的技术，包括 Keil μVision4 集成开发环境与 RealView 编译器。支持 ARM7、ARM9、Cortex-M3 和最新的 Cortex-M4 核处理器，自动配置启动代码，集成 Flash 烧写模块，强大的 Simulation 设备模拟，性能分析等功能，与 ARM 之前的工具包 ADS 等相比，RealView 编译器的最新版本可将性能改善超过 20%。

最新发布的 Keil μVision4 引入灵活的窗口管理系统，使开发人员能够使用多台监视器，并提供了视觉上的表面对窗口位置的完全控制的任何地方。新的用户界面可以更好地利用屏幕空间和更有效地组织多个窗口，提供一个整洁，高效的环境来开发应用程序。同时，新版本支持更多最新的 ARM 芯片（包括飞思卡尔公司推出的内核为 Cortex-M4 的 Kinetis 系列），还添加了一些其他新功能。ARM 公司发布最新集成开发环境 RealView MDK 开发工具中集成了最新版本的 Keil uVision4，其编译器、调试工具实现与 ARM 器件的最完美匹配。

由于 Kinetis 系列采用 ARM 公司优秀的 Cortex-M4 核，我们选择采用 Keil μVision4 作为程序开发调试软件。程序下载由 Keil4 和 JLink 下载器完成。

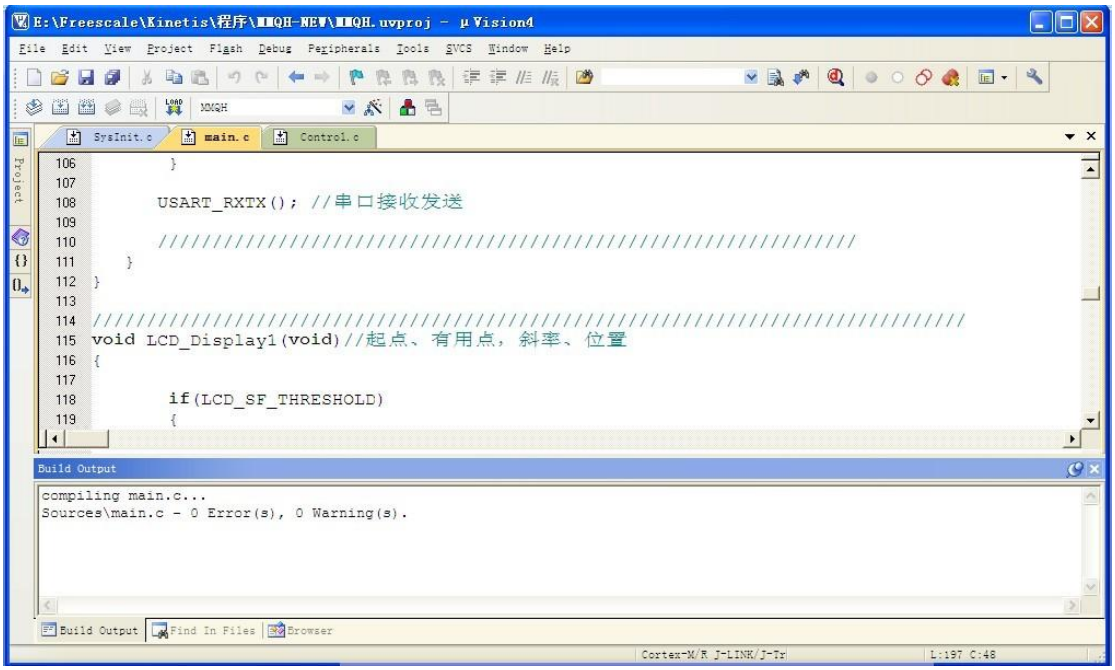


图 6-1 Keil4 编程界面

6.2 J-LINK 仿真器

J-Link 是 SEGGER 公司为支持仿真 ARM 内核芯片推出的 JTAG 仿真器。配合 IAR EWAR，ADS，KEIL，WINARM，RealView 等集成开发环境支持所有 ARM7/ARM9/ARM11/Cortex-Mx 内核芯片的仿真，通过 RDI 接口和各集成开发环境无缝连接，操作方便、连接方便、简单易学，是学习开发 ARM 最好最实用的开发工具。J-LINK 仿真器 V8 版，其仿真速度和功能远非简易的并口 WIGGLER 调试器可比。通过 J-LINK 仿真器，可以实现 Kinetis 开发编程、下载、在线调试一站式完成。特别是其强大的在线调试功能在调试智能车过程非常方便。

图 6-2 J-LINK 仿真器

6.3 上位机调试软件设计

为了更好地掌握智能车在行驶中的各种状态，以制定更好地控制方案，我们用 LabvIEW 编写了一个智能车上位机,用以对智能车进行实时的监控与控制。

LabVIEW 是一种程序开发环境，由美国国家仪器（NI）公司研制开发的，类似于 C 和 BASIC 开发环境，但是 LabVIEW 与其他计算机语言的显著区别是：其他计算机语言都是采用基于文本的语言产生代码，而 LabVIEW 使用的是图形化编辑语言 G 编写程序，产生的程序是框图的形式。LabVIEW 提供很多外观与传统仪器（如示波器、万用表）类似的控件，可用来方便地创建用户界面。使用图标和连线，可以通过编程对用户界面上的对象进行控制。这就是图形化源代码，又称 G 代码。LabVIEW 的图形化源代码在某种程度上类似于流程图，因此又被称作程序框图代码。

本上位机通过串口接收模型车上传的数据帧，然后根据数据帧的类型进行曲线描画等工作。这样就大大方便了调试者观察车子的实时状态。同时，它还提供了简单的上位机对模型车的控制功能，实现远程监控与远程控制。

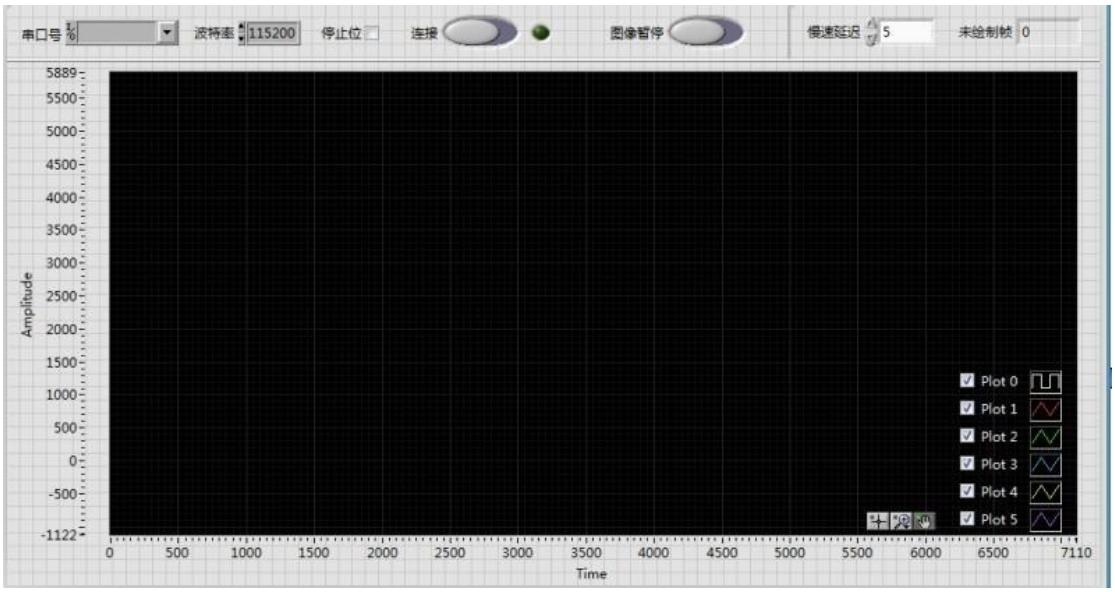


图 6-2-1 上位机界面

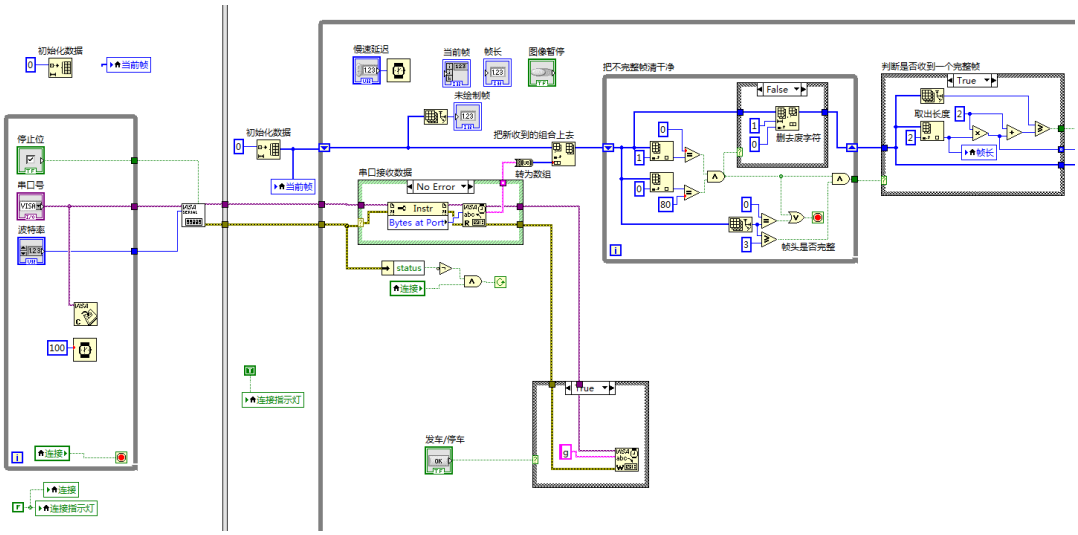


图 6-2-2 上位机部分程序图

6.4 蓝牙无线通信在智能车调试中的应用

由于模型车在跑道上实际运行时都处于高速运行状态，调试人员不可能随时跟着车跑进行观察。因此，调试人员要在上位机了解其运行时的参数变化情况。本设计选取了方便易用的蓝牙传输方式传送数据给远程监控系统，调试人员就能在 PC 端监控机器人的运行，方便地进行离线数据分析。蓝牙模块发挥的作用是：智能车的控制程序把图像、转向给定、车速给定、车体位置、车速、转向等数据通过蓝牙模块发送到 PC 端的监控软件上，充当通信介质。此蓝牙模块可在 20 米内可靠传输数据。

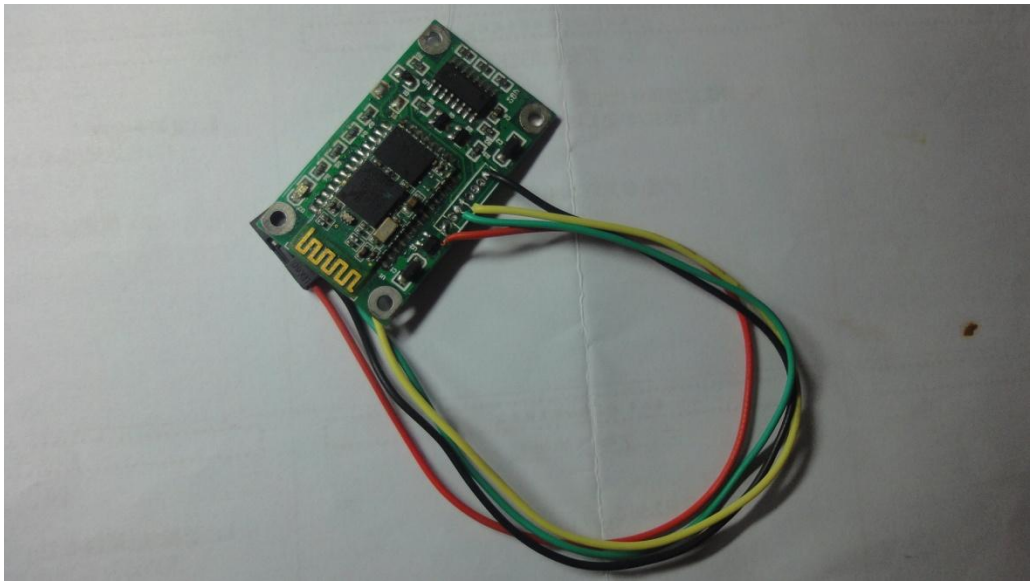


图 6-3 蓝牙模块

6.5 TFT 显示屏的应用

车载显示屏方便地提供了人机交互功能，调试人员可以直接通过显示屏第一时间了解模型车拍摄的图像和提取的路径以及各类参数，在某些场合提供了方便。以下是我们设计的显示屏（车载）：

图 6-5 TFT 显示屏模块

第七章 模型车主要技术参数说明

表 7-1 模型车主要技术参数

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）	280*230*290
车模轴距/轮距（毫米）	150
车模平均电流（匀速行驶）（毫安）	1500
电路电容总量（微法）	1700
传感器种类及个数	ov7620 一个，光电编码器一个，激光接收头一个
新增加的电机个数	0
赛道信息检测空间精度（毫米）	3
赛道信息检测频率（次/秒）	60
主要集成电路种类/数量	MK60DN512 一片

车模重量（带有电池）（千克）	1.2

第八章 总结

技术报告详细介绍了基于飞思卡尔公司推出的 Kinetis 系列 K60 微处理器制作竞速智能车的整个设计流程。其中主要包括机械调整、硬件电路设计和软件算法设计三个方面内容。机械上主要介绍了如何调整前轮悬挂以获得平稳的直线行驶以及性能较高的拐弯效果，摄像头的安装方法，以及兼容快速反应和足够扭矩的舵机安装方法等。硬件电路则介绍了非常重要的稳压电路、采用 DMA 采集图像的方案、单片机选型以及 CMOS 摄像头与 CCD 摄像头比较的优与劣，另外，采用 TFT 实时显示模型车的状况大大方便了调试以及赛车比赛。软件算法上，详细介绍了图像采集的算法流程，图像二值化阈值的提取方法，可靠的路径边缘提取的边缘检测法，此外，还对几个相对比较特殊的十字路口赛道和终点线识别算法进行了详细的介绍。最后，对我们几个月以来调试智能车的软件、工具做了简要的介绍。

经过华南赛区的比赛检验，我们认为以上方案有足够的竞争能力。但是，由于我们相对薄浅的专业知识和相对缺乏的社会经验，这个制作方案仍然存在着些不足。例如机械上还是比较粗糙，没能做到尽量极致的程度；模型车图像中的路径特征提取还可以做得更精准，行驶的路径还可以更好，速度控制还可以更合理。当然，智能车没有绝对的完美无暇，精益求精正是飞思卡尔智能车比赛的魅力所在。

半年的备赛是一个艰苦的过程，同时也是个快乐的过程。因为我们在这个过程一直处于发现问题和解决问题的无限循环中。每次成功解决一个问题都足以让我们高兴得手舞足蹈。所以，这是一个不断收获的过程，我们收获了坚强的意志，必胜的信念，勤劳耐苦的心，高效的动手解决问题能力以及强大的团队协作能力等等。此外，这个比赛让我们有机会开阔自己的眼界，培养了我们积极阳光的公平竞争精神，同时也让我们有机会结交更多朋友。我们相信，智能车备赛是我们人生旅途中的一笔宝贵财富，一定会让我们铭记在心，值得我们以后细细回味。

致谢

在我们几个月的备赛过程中，学校教务处和自动化学院的相关领导、老师在经费和场地上给予了大力的支持。在此特别向给予我们帮助的老师们、同学们表示感谢。同时，感谢大赛组委会组织了一项这么有意义的比赛。

参考文献

[1] 卓晴,黄开胜,邵贝贝. 学做智能车[M]. 北京：北京航空航天大学出版社. 2007.

[2] 谭浩强著. C 程序设计[M]. 北京：清华大学出版社，2003.

[3] 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.

[4] 华成英,童诗白. 模拟电子技术基础[M]. 北京: 高等教育出版社. 2006.

[5] 阎石. 数字电子技术基础[M]. 北京: 高等教育出版社. 2006.

[6] Curtis D.Johnson. 过程控制仪表技术[M]. 北京: 清华大学出版社. 2009.

[7] 韩峻峰，李玉惠. 模糊控制技术[M]. 重庆. 重庆大学出版社. 2003.

[8] 张化光，何希勤. 模糊自适应理论及应用[M]. 北京：北京航空航天大学出版社. 2002.

[9] 历风满. 数字 PID 控制算法的研究. 辽宁大学学报 自然科学版 第 32 卷 第 4 期 2005.

[10] 王超艺. 基于 Cortex-M4 内核的 Kinetis 微控制器的应用研究.2011.

[11] Freescale K60P100M100SF2 Rev. 4, 3/2011.

[12] Freescale K60P100M100SF2RM Rev. 6, Nov 2011.

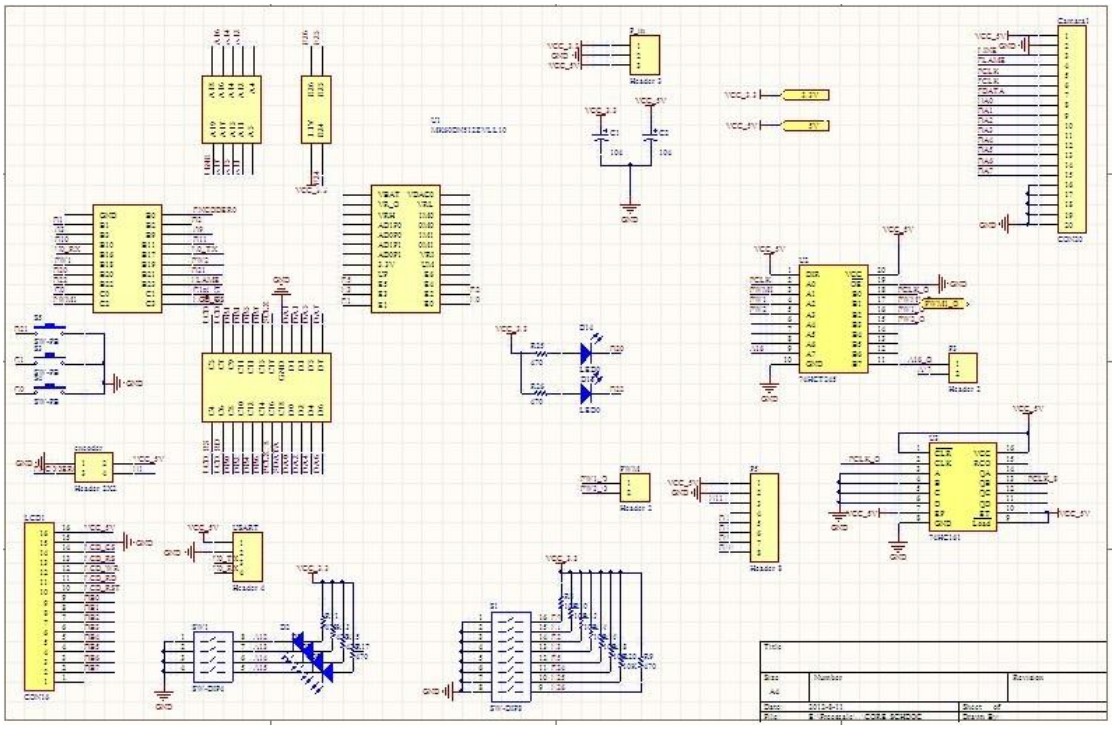
[13] Freescale Kinetis Peripheral Module Quick Reference Rev 1, 03/2012.

[14] http://www.smartcar.au.tsinghua.edu.cn/web/rules.html

[15] http://modify.modiauto.com.cn

附录

附录 A 电路板原理图



附录 B 部分源程序

Main.c

```
#include "Includes.h"   /* common headers  */
////////////////////////////////////////////////////////////////
///
INT16U uiCount=0;
INT8U ucRealTimer=0;
void LCD_Display1(void);//起点、有用点，斜率、位置
void LCD_Display2(void);//舵机，电机
void USART_RXTX(void); //串口接收发送
////////////////////////////////////////////////////////////////
////
int main(void)
{
   u16 uiI=0;
   SysInit();
   while(1)
   {
     if(LCD_SF_IMAGE)
     {
        if(FrameShowEnable==1)
        {
            FrameShowEnable=0; //清除标志
           LCD_ShowImage(5,10,cPX,cPY,ucImage_buf[0]);
        }
     }
     else if(LCD_SF_2VALUE )
      {               if(FrameShowEnable==1)
       {
```

```
                FrameShowEnable=0;   //清除标志
    //          if(ucAutoLockFlag==0)
                g_ucThreshold=ucGet_2value_threshold(ucImage_buf); //获取阀值

                 Image_2value(ucImage_buf);  //二值化图像
                 LCD_ShowImage(5,70,cPX,cPY,ucImage_Temp[0]);
              }
           }
           else
           {
             g_ucThreshold=ucGet_2value_threshold(ucImage_buf); //获取阀值

              SearchCenterLine(); //搜索路线
              for(uiI=0;uiI<cPY;uiI++)ucRowEndFlag[uiI]=0;
              if(LCD_SF_EDGELINE)LCD_ShowEdgeLine2(5,     130,     cPX,
cPY,&s_LE,&s_RE);

              ImageAdjust2(&s_LE);//图像矫正
              ImageAdjust2(&s_RE);
              if(LCD_SF_ADJLINE)LCD_ShowEdgeLine3(5,      190,     cPX,
cPY,&s_LE,&s_RE);
                      GetCenterLine();    //获取中心线
            Get_Pos_Slope(&s_CL);

////////////////////////////////////////////////////////////
              if(LCD_SF_CENTERLINE)  LCD_ShowCenterLine2(5,  250,  cPX,
cPY,&s_CL);
            CenterLineGetTurnPoint(&s_CL);
```

```
//////////////////////////////////////////////////////////////
                if(LCD_SF_VAL==0||LCD_SF_VAL>7)
                {
                                if(s_CL.ucUsefulPoint)
                 {
                   LoadShapeProcess();
                 }
                 LCD_Display1();//起点、有用点，斜率、位置
                 LCD_Display2();//舵机，电机
              }
            }
          USART_RXTX(); //串口接收发送

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

       }
    }
    //////////////////////////////////////////////////////////////
///////////////
    void LCD_Display1(void)//起点、有用点，斜率、位置
    {
           if(LCD_SF_THRESHOLD)
           {
              LCD_ShowNum(200,160,g_ucThreshold,3,16);
           }
           if(LCD_SF_CENTERORIGN)
           {
           /*
LCD_ShowNum(120,0,s_LE.cLine[s_LE.ucStartPoint]+127,3,16);
```

```
                LCD_ShowNum(120,20,s_LE.ucStartPoint,2,16);
                LCD_ShowNum(120,40,s_LE.ucUsefulPoint,2,16);
             //    LCD_ShowNum(120,60,s_LE.ucTurnPointCnt,2,16);

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

LCD_ShowNum(150,0,s_RE.cLine[s_RE.ucStartPoint]+127,3,16);
                LCD_ShowNum(150,20,s_RE.ucStartPoint,2,16);
                LCD_ShowNum(150,40,s_RE.ucUsefulPoint,2,16);
            //     LCD_ShowNum(150,60,s_RE.ucTurnPointCnt,2,16); */

LCD_ShowNum(150,0,s_CL.cLine[s_CL.ucStartPoint]+127,3,16);
            //     LCD_ShowNum(150,20,s_CL.ucStartPoint,2,16);
                LCD_ShowNum(150,20,s_CL.ucStartPoint,3,16);

                LCD_ShowNum(150,40,s_CL.ucUsefulPoint,2,16);

LCD_ShowNum(150,60,s_CL.cLine[s_CL.ucUsefulPoint]+127,3,16);
                        }
            if(LCD_SF_SLOPE_POS)
            {
                 LCD_ShowNum(120,100,s_CL.iWholeslope+127,3,16);
                 LCD_ShowNum(120,120,s_CL.iWholeposition+127,3,16);
                  LCD_ShowNum(200,80,iAngle,4,16);

            }
    }
    void LCD_Display2(void)//舵机，电机
    {
            if(LCD_SF_MOTORSPEED)
```

```
          {
            LCD_ShowNum(180,100,iSpeedSet+1000,4,16);
         }
              if(LCD_SF_ADVALUE){
            LCD_ShowNum(180,120,uiEncoder_Pulse,4,16);
            }                          if(LCD_SF_MOTORSPEED)
          {
     //       LCD_ShowNum(180,140,iSpeedOut/10+1000,4,16);
         }
}
void USART_RXTX(void) //串口接收发送
{       uiCount++;
        if(uiCount==3&&uc_SF_Advalue=='1')
        {
                    SCI_SendData(uiEncoder_Pulse+1000,'o');
        }       if(uiCount==4&&uc_SF_SpeedSet=='1')
        {
                  SCI_SendData(iSpeedSet+1000,'s');
        }
        if(uiCount>5&&uc_SF_SpeedOut=='1')
        {           SCI_SendData(iSpeedOut/10+1000,'a');
          //  SCI_SendData(iAngle-1000,'o');
        }
        if(Imageflag==1)
        {
           Imageflag=0;
           SendImage(ucImage_buf);
         //   Regurate();
        }
             if(uiCount>5)
```

```
            uiCount=0;
    }
```

ImageProcess.c

```
    #include "Includes.h"
    unsigned char ucImage_buf[cPY][cPX]={{0}};
    struct s_Line s_LE={0,0,0,0,0,{0,0},{0},{0},{0}},
                  s_RE={0,0,0,0,0,{0,0},{0},{0},{0}},
                  s_CL={0,0,0,0,0,{0,0},{0},{0},{0}};
    INT16U uiHist[256]={0};
    INT8U  ucThreshold=0;              //二值化阀值
    INT8U ucThreshold_Near=0;             //近半屏阀值
    INT8U ucThreshold_Far=0;              //远半屏阀值
    INT8U  ucMaxDotOffset=0;   //两行点最大偏移限制

    INT8U ucSearchFlag[2]={0};          //允许搜索标志位(十字交叉补线
用)
    INT8U ucGet_2value_threshold(INT8U ucImage[][cPX]){
      INT8U  ucX,ucY;
      INT8U  ucN;
      INT16U uiPeak_while=0;//直方图峰值
      INT8U  ucThreshold_high,ucThreshold_low;//峰值对应的亮度
      INT8U  ucThreshold_scale;
          ucThreshold_scale=cTotalThresholdRate;

      for(ucN=0;ucN<255;ucN++)
      {
          uiHist[ucN]=0;
      }
      for(ucY=cSAMPLE_GAP;ucY<cPY-cSAMPLE_GAP;ucY++)
      {     //求直方图
```

```
        for(ucX=cSAMPLE_GAP;ucX<cPX-cSAMPLE_GAP;ucX+=2)
        {
         ucN=ucImage[ucY][ucX];
         uiHist[ucN]+=1;
        }
      }
      for(ucN=0;ucN<127;ucN++) {        //求第一个峰值
        if(uiPeak_while<uiHist[ucN]) {
          uiPeak_while=uiHist[ucN];
          ucThreshold_low=ucN;
        }
      }
      for(ucN=127;ucN<255;ucN++)
      {       //求白色峰值
        if(uiPeak_while<uiHist[ucN])
        {
          uiPeak_while=uiHist[ucN];
          ucThreshold_high=ucN;
        }
      }
      ucThreshold=(INT8U)(ucThreshold_high*ucThreshold_scale/10);
      ucThreshold_Near=(INT8U)(ucThreshold*cNearThresholdRate/10);//
近半屏阀值
      ucThreshold_Far=(INT8U)(ucThreshold*cFarThresholdRate/10);// 远
半屏阀值
      return ucThreshold;
    }
    //上下半屏两阀值 2 值化
    void Image_2value(INT8U ucImage[][cPX])
    {
```

```
INT8U ucX,ucY;

//下 1/2 屏(近处)2 值化
for(ucY=0;ucY<cPY/2;ucY++)
{
  for(ucX=0;ucX<cPX;ucX++)
  {
    if(ucImage[ucY][ucX]<ucThreshold_Near)
    {
      ucImage[ucY][ucX]=ucCCD_BLACK ;
    }
    else
    {
      ucImage[ucY][ucX]=ucCCD_WHITE ;
    }
  }
}
//上 1/2 半屏(远处)2 值化
for(ucY=cPY/2;ucY<cPY;ucY++)
{
  for(ucX=0;ucX<cPX;ucX++)
  {
    if(ucImage[ucY][ucX]<ucThreshold_Far)
    {
      ucImage[ucY][ucX]=ucCCD_BLACK ;
    }
    else
    {
      ucImage[ucY][ucX]=ucCCD_WHITE ;
    }
```

```
        }
      }

}
void SearchCenterLine(void) //搜索中心线
{
    INT8U ucI=0,ucJ=0;
    ////////////////////////////////////////////////////////////
    for(ucI=0;ucI<cPY;ucI++)  //从起点往上搜边界线
    {       if(s_LE.ucUsefulPoint==cPY_1)
        {
                    SearchEdgeLine(&s_LE,ucI); //搜左边界
        }
        if(s_RE.ucUsefulPoint==cPY_1)
        {
                     SearchEdgeLine(&s_RE,ucI); //搜右边界
        }
        ////////////////////////////////////////
        if(cSHIZICrossFlag>0)//如果十字交叉标志位有效
        {
            LCD_ShowNum(120,100,cSHIZICrossFlag,1,16);

            ucI=cManualAddPoints(); //手工补点
            LCD_ShowNum(180,100,ucI,2,16);
            cSHIZICrossFlag=-1;
        }

     ////////////////////////////////////////////////////////////
    }
    for(ucI=s_LE.ucStartPoint;ucI<=s_LE.ucUsefulPoint;ucI++)
```

```
    {

        if(s_LE.cLine[ucI]<0||s_LE.cLine[ucI]>79)
        {
            LCD_ShowNum(180,120,1,1,16);
            LCD_ShowNum(180,140,s_LE.cLine[ucI],3,16);
            for(ucJ=0;ucJ<cPY;ucJ++)
           {
              SCI_SendData1(s_LE.cLine[ucJ]);
           }


        }
    }

    for(ucI=s_RE.ucStartPoint;ucI<=s_RE.ucUsefulPoint;ucI++)
    {

        if(s_RE.cLine[ucI]<0||s_LE.cLine[ucI]>79)
        {
            LCD_ShowNum(180,120,2,1,16);
            LCD_ShowNum(180,140,s_RE.cLine[ucI],3,16);
            for(ucJ=0;ucJ<cPY;ucJ++)
           {
              SCI_SendData1(s_RE.cLine[ucJ]);
           }
        }
    }

}
```

```
char cOneLineSearchPoint(struct s_Line *pts,INT8U ucI,INT8U ucStartx,INT8U ucEndx,INT8U ucThreshold)
{  //边界法

   INT8U ucJ=0;
   INT8U ucWhiteCnt=0,ucBlackCnt=0;
   ////////////////////////////////////////////////////////
   if(pts==&s_LE) //搜左边界
   {

         for(ucJ=ucEndx;ucJ>ucStartx;ucJ--)
         {
            if(ucImage_buf[ucI][ucJ]>=ucThreshold)//白点
            {
                ucWhiteCnt++;
                ucBlackCnt=0;
            }
            else// if(ucImage_buf[ucI][ucJ]<ucThreshold) //黑点
            {
               ucBlackCnt++;
               if(ucBlackCnt>1&&ucWhiteCnt>1)
                  return ucJ+2;
            }
         }
         if(ucWhiteCnt>ucEndx-ucStartx-3)//几乎全部为白
              return cAllWhite;
         else if(ucWhiteCnt<3) //几乎全部为黑
              return cAllBlack;
         else
              return cNoPoint;//没找到合适点
```

```
}
else if(pts==&s_RE) //搜右边界
{

        for(ucJ=ucStartx;ucJ<ucEndx;ucJ++)
        {
           if(ucImage_buf[ucI][ucJ]>=ucThreshold)//白点
           {
               ucWhiteCnt++;
               ucBlackCnt=0;
           }
           else// if(ucImage_buf[ucI][ucJ]<ucThreshold) //黑点
           {
              ucBlackCnt++;
              if(ucBlackCnt>1&&ucWhiteCnt>1)                {

                 return ucJ-2;
              }
           }
        }
        if(ucWhiteCnt>ucEndx-ucStartx-3)//几乎全部为白
             return cAllWhite;
        else if(ucWhiteCnt<3) //几乎全部为黑
             return cAllBlack;
        else
             return cNoPoint;//没找到合适点


}
```

```
    }

    void ProcessParallelSearch(struct s_Line *pts,INT8U ucRowNum,INT8U*
pt_ucAutoAddCtr)
    {
         INT8U ucI=0,ucJ=0;
         INT8U ucCounter=0;
         INT8S cTopColor[2]={0,0};//0 为左，1 为右


////////////////////////////////////////////////////////////////////
////////////////////////////
         if(ucRowNum<cPY_20)
             ucMaxDotOffset=5;   //两行点最大偏移限制
         else
             ucMaxDotOffset=4;   //两行点最大偏移限制

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

         if(pts==&s_LE) //搜左边界
         {
           if(pts->cLine[ucRowNum]<cNoPoint)  //找不到
           {
             cTopColor[0]=pts->cLine[ucRowNum];//暂存下颜色
              if(cTopColor[0]==cAllWhite)          //全白点计数
              {
                  pts->cTopColor[0]++;
              }
              else
```

```
            {
                pts->cTopColor[0]=0;
                ucSearchFlag[0]=0;//停止补点
            }
                    pts->cLine[ucRowNum]=cNoPoint;
        }
                if(pts->cLine[ucRowNum]>cNoPoint)   //搜到
        {
            pts->cTopColor[0]=0;
            ucSearchFlag[0]=0;
         }

        if(pts->cLine[ucRowNum]==cNoPoint) //没搜到合适点

{       ////////////////////////////////////////////////////////////
////////////////////////////////////

pts->cLine[ucRowNum]=cVerticalSearchPoint(pts,ucRowNum);

if(uiAbs1(pts->cLine[ucRowNum],pts->cLine[ucRowNum-1])>ucMaxDotOffse
t)
            {
                pts->ucUsefulPoint=ucRowNum;
            }
        //    SCI_TxChar('0');
        //    SCI_SendData1(pts->cLine[ucRowNum]);

////////////////////////////////////////////////////////////////////
////////////////////////////////
                    if(((ucRowNum)-(pts->ucStartPoint))<4          ||
```

```
ucSearchFlag[0]==1)
                pts->cLine[ucRowNum]=pts->cLine[ucRowNum-1];//找不到
             else
             {

pts->cLine[ucRowNum]=pts->cLine[ucRowNum-2]+(pts->cLine[ucRowNum-2]-
pts->cLine[ucRowNum-3]);//线性补点
                      //    }

            if(pts->ucUsefulPoint==cPY_1)
            {
               //    ucCounter=10-(ucRowNum)>>3;
                  if((*pt_ucAutoAddCtr)<4)
                  {
                        (*pt_ucAutoAddCtr)++;//计数+1
                  }
                   else
                   {
                        pts->ucUsefulPoint=ucRowNum;//-ucCounter;  //
有用点最大纵坐标
                                          }
            }
          }
               }
      else if(pts==&s_RE) //搜右边界
      {
          if(pts->cLine[ucRowNum]<cNoPoint)    //找不到
          {
             cTopColor[0]=pts->cLine[ucRowNum];//暂存下颜色
             if(cTopColor[0]==cAllWhite)
```

```
            {
                pts->cTopColor[0]++;
                   }
            else
            {
                pts->cTopColor[0]=0;
                ucSearchFlag[1]=0;//停止补点
            }
                      pts->cLine[ucRowNum]=cNoPoint;
        }
        if(pts->cLine[ucRowNum]>0)   //搜到
        {
            pts->cTopColor[0]=0;
            ucSearchFlag[1]=0;
       //   else
       //        *pt_ucAutoAddCtr=0;//计数清零
        }

         if(pts->cLine[ucRowNum]==cNoPoint)  // 没 搜 到 合 适 点
//////////////   <= cNoPoint
         {

////////////////////////////////////////////////////////////////////
////////////////////////////
            pts->cLine[ucRowNum]=cVerticalSearchPoint(pts,ucRowNum)

if(uiAbs1(pts->cLine[ucRowNum],pts->cLine[ucRowNum-1])>ucMaxDotOffse
t)
            {
                pts->ucUsefulPoint=ucRowNum;
```

```
                }
        //        SCI_TxChar('R');
        //       SCI_SendData1(pts->cLine[ucRowNum]);

////////////////////////////////////////////////////////////////////
////////////////////////////////
                if(((ucRowNum)-(pts->ucStartPoint))<4                ||
ucSearchFlag[1]==1 )
                 pts->cLine[ucRowNum]=pts->cLine[ucRowNum-1];//找不到
              else

{                  pts->cLine[ucRowNum]=pts->cLine[ucRowNum-2]+(pts->c
Line[ucRowNum-2]-pts->cLine[ucRowNum-3]);//线性补点

      //        }
                        if(pts->ucUsefulPoint==cPY_1)
             {
                //    ucCounter=10-(ucRowNum)>>3;
                   if((*pt_ucAutoAddCtr)<4)
                   {
                         (*pt_ucAutoAddCtr)++;//计数+1
                   }
                    else
                    {
                         pts->ucUsefulPoint=ucRowNum;//-ucCounter;  //
有用点最大纵坐标
                                                  }
               }
           }
       }
```

```
//////////////////////////////////////////////////////////////////////
/////////////
       /////////校正前连续记录四个点
        pts->cLine[ucRowNum-4]=pts->cLine[ucRowNum-3];
        pts->cLine[ucRowNum-3]=pts->cLine[ucRowNum-2];
        pts->cLine[ucRowNum-2]=pts->cLine[ucRowNum-1];
        pts->cLine[ucRowNum-1]=pts->cLine[ucRowNum];
            if(LCD_SF_CENTERLINE)
        {
              if(ucRowNum==pts->ucUsefulPoint)
          {

pts->cLine[ucRowNum]=cAdjustTable[ucRowNum][pts->cLine[ucRowNum]];//
矫正有用点

pts->cLine[ucRowNum-1]=cAdjustTable[ucRowNum-1][pts->cLine[ucRowNum-
1]];
        }
        ////////////////////////////////////////////////////
        if(ucRowNum>=pts->ucStartPoint+2)

pts->cLine[ucRowNum-2]=cAdjustTable[ucRowNum-2][pts->cLine[ucRowNum-
2]]; //前前个点矫正
        if(ucRowNum>=pts->ucStartPoint+8)
           cGet_TurnPoint(pts,ucRowNum-3);
            if(cSHIZICrossFlag==0)
           cJudge_SHIZICross(pts);
           //
    }
```

```
//////////////////////////////////////////////////////////////////////
/////////////
    }
    INT16U uiAbs_INT16(INT16S iNum)
    {
       if(iNum<0)return -iNum;
       else     return iNum;
    }

    INT16U uiAbs1( int iX1,int iX2)
    {
        if(iX1>iX2)
        {
            return (iX1-iX2);
        }
        else
        {
             return (iX2-iX1);
        }
    }
    INT8S cFindAttri_INT16(INT16S iNum)
    {   if(iNum<0)
          return cNegative;
       else if(iNum>0)
       {
         return cPositive;
       }
       else
       {
         return cZero ;
```

```
        }
         }
    char     cManualAddPoints(INT8S     cCenter,INT8U     ucRowNum,INT8S
cOffset,INT8U ucInRowNum) //手工补点
    {
        INT8U ucI=0,ucJ=0,ucM=0,ucN=0;
        INT8U ucThreshold=0;
        INT8U ucStartx=0,ucEndx=cPX;     //黑线搜索区起末 x 坐标
        INT8U ucX_half_range=30;       //黑线搜索区 x 坐标半宽

        INT8S cResult1=0,cResult2=0,cNum1=0,cNum2=0,cError=0;
        INT16S iSum=0;

        cError= cOffset/2;

        for(ucJ=ucRowNum+3;ucJ<cPY-5;ucJ+=2)
        {
            if(ucJ>=cPY_31)
                 ucThreshold=ucThreshold_Far;
            else
                 ucThreshold=ucThreshold_Near;


    ucStartx=(cCenter-ucX_half_range<=0?0:(INT8U)(cCenter-ucX_half_ra
nge));
            ucEndx
=(cCenter+ucX_half_range>=cPX?cPX_1:(INT8U)(cCenter+ucX_half_range))
;
```

```
cResult1=cOneLineSearchPoint(&s_LE,ucJ,ucStartx,(INT8U)cCenter,ucThr
eshold);

cResult2=cOneLineSearchPoint(&s_RE,ucJ,(INT8U)cCenter,ucEndx,ucThres
hold);

            if(cResult1>0&&cResult2>0)
            {
                 s_LE.cLastLine[0]=cResult1;
                 cResult1=cAdjustTable[ucJ][cResult1];//点矫正
                if(cSHIZICrossType==5||cSHIZICrossType==6)//特殊情况，
底下一片白
                {
                    for(ucN=0;ucN<ucJ;ucN++)
                        s_LE.cLine[ucN]=cResult1;
                    ////////////////////////////////
                    s_LE.ucStartPoint=0;
                }
                else
                {
                     cNum1=                cResult1                -
s_LE.cLine[s_LE.ucUsefulPoint-2];
                     cNum2= ucJ       - (s_LE.ucUsefulPoint-2);

                     for(ucN=s_LE.ucUsefulPoint-2;ucN<ucJ;ucN++)   //
逐点补点
                     {
                         iSum+=cNum1;

s_LE.cLine[ucN]=s_LE.cLine[s_LE.ucUsefulPoint-2]+iSum/cNum2;
```

```
                }
            }
            s_LE.ucUsefulPoint=cPY_1;
            s_LE.ucLastSearchPoint=ucJ;
            s_LE.ucAutoAddCtr=0;
            s_LE.ucSuspendPoint=cPY_1;
            s_LE.cTrend[0]=0;
            s_LE.cLastLine[1]=s_LE.cLastLine[0];
            s_LE.cLastLine[2]=s_LE.cLastLine[0];
            s_LE.cLastLine[3]=s_LE.cLastLine[0];

////////////////////////////////////////////////////////////
            iSum=0;//清零
            s_RE.cLastLine[0]=cResult2;
            cResult2=cAdjustTable[ucJ][cResult2];//点矫正
            if(cSHIZICrossType==5||cSHIZICrossType==6)//特殊情
况，底下一片白
            {
               for(ucN=0;ucN<ucJ;ucN++)
                   s_RE.cLine[ucN]=cResult2;
               ////////////////////////////////
               s_RE.ucStartPoint=0;
            }
            else
            {
                cNum1=                    cResult2          -
s_RE.cLine[s_RE.ucUsefulPoint-2];
                cNum2=  ucJ      - (s_RE.ucUsefulPoint-2);
```

```
                    for(ucN=s_RE.ucUsefulPoint-2;ucN<ucJ;ucN++)   //
逐点补点
                    {
                        iSum+=cNum1;

s_RE.cLine[ucN]=s_RE.cLine[s_RE.ucUsefulPoint-2]+iSum/cNum2;

                    }
                }
                s_RE.ucUsefulPoint=cPY_1;
                s_RE.ucLastSearchPoint=ucJ;
                s_RE.ucAutoAddCtr=0;
                s_RE.ucSuspendPoint=cPY_1;
                s_RE.cTrend[0]=0;
                s_RE.cLastLine[1]=s_RE.cLastLine[0];
                s_RE.cLastLine[2]=s_RE.cLastLine[0];
                s_RE.cLastLine[3]=s_RE.cLastLine[0];

//////////////////////////////////////////////////////////////////
                return ucJ;

            }
            cCenter=cCenter+cError;//开窗偏移

            if(cCenter>cPX) //限幅
               cCenter=cPX;
            else if(cCenter<0)
               cCenter=0;
        }
```

```
    return ucInRowNum;

}
```

Control.c

```
#include "Control.h"
#include "pit.h"
#include "ftm.h"
#include "gpio.h"
#include "Settings.h"
INT8U  ucRoadType=cLongStraightRoad;  //道路形状
void WholeLoadProcess(void)
{
    int NewSlope=0,OldSlope=0;
    NewSlope=s_CL.iWholeslope;
    if(s_MoveMnr.Ctrl_Cnt<1)
    {
        OldSlope=s_MoveMnr.Slope[9];
    }
    else
    {
        OldSlope=s_MoveMnr.Slope[s_MoveMnr.Ctrl_Cnt-1];
    }

    s_MoveMnr.D_Slope[s_MoveMnr.Ctrl_Cnt]=NewSlope-OldSlope;
    s_MoveMnr.Slope[s_MoveMnr.Ctrl_Cnt]=NewSlope;
    s_MoveMnr.Tendency-=s_MoveMnr.D_Sign[s_MoveMnr.Ctrl_Cnt];

if(s_MoveMnr.D_Slope[s_MoveMnr.Ctrl_Cnt]>10||s_MoveMnr.D_Slope[s_Mov
eMnr.Ctrl_Cnt]<-10)
    {
```

```
            if(uiAbs_INT16(NewSlope)>=uiAbs_INT16(OldSlope))
            {
                s_MoveMnr.D_Sign[s_MoveMnr.Ctrl_Cnt]=1;
                s_MoveMnr.Tendency+=1;
            }
            else
            {
                s_MoveMnr.D_Sign[s_MoveMnr.Ctrl_Cnt]=-1;
                s_MoveMnr.Tendency+=-1;
            }
        }
        else
        {
            s_MoveMnr.D_Sign[s_MoveMnr.Ctrl_Cnt]=0;
        }

        s_MoveMnr.Ctrl_Cnt++;
        if(s_MoveMnr.Ctrl_Cnt>=10)s_MoveMnr.Ctrl_Cnt=0;


////////////////////////////////////////////////////////////////////
//////////////////
    //    if(s_MoveMnr.Tendency<-5)
    //    {
    //         Error_Speedpoint=Error_Speedpoint/2;
    //    }
    //    else
        LCD_ShowNum(180,40,s_MoveMnr.Tendency+10,3,16);
        LCD_ShowNum(180,60,Error_Speedpoint+1000,4,16);
        if(s_MoveMnr.Tendency<0)
```

```
        {
Error_Speedpoint=Error_Speedpoint*(10-s_MoveMnr.Tendency)/10;
        }
        else if(s_MoveMnr.Tendency<3)
        {

Error_Speedpoint=Error_Speedpoint*(12-s_MoveMnr.Tendency)/10;
        }
        else if(s_MoveMnr.Tendency<11)
        {

Error_Speedpoint=Error_Speedpoint*(10-s_MoveMnr.Tendency)/15;
        }
       }
    void GetCenterLine(void)  //获取中心线
    {
       INT8U ucStarty=0,ucEndy=0,ucI;
      //有一边丢线
        if(                (s_LE.ucStartPoint==cPY_1)                ||
((s_LE.ucUsefulPoint-s_LE.ucStartPoint)<8))
        {
            s_LE.ucStartPoint=0;
            s_LE.ucUsefulPoint=0;
         }
            if((s_RE.ucStartPoint==cPY_1)                              ||
((s_RE.ucUsefulPoint-s_RE.ucStartPoint)<8))
        {
            s_RE.ucStartPoint=0;
            s_RE.ucUsefulPoint=0;
        }
```

```
        if( s_RE.ucUsefulPoint<s_LE.ucStartPoint)
        {
            s_RE.ucUsefulPoint=0;
        }

        if(s_LE.ucUsefulPoint<s_RE.ucStartPoint)
        {
            s_LE.ucUsefulPoint=0;
        }
             if(s_LE.ucUsefulPoint!=0&&s_RE.ucUsefulPoint!=0)        //
两边线都存在
        {
             if(s_LE.ucStartPoint>=s_RE.ucStartPoint)
            {
               ucStarty=s_LE.ucStartPoint;
               s_CL.ucStartPoint=s_RE.ucStartPoint;
            }
            else
            {
               ucStarty=s_RE.ucStartPoint;
               s_CL.ucStartPoint=s_LE.ucStartPoint;
            }
                    if(s_LE.ucUsefulPoint<=s_RE.ucUsefulPoint)
            {
                ucEndy=s_LE.ucUsefulPoint;
                s_CL.ucUsefulPoint=s_RE.ucUsefulPoint;
            }
            else
            {
```

```
            ucEndy=s_RE.ucUsefulPoint;
            s_CL.ucUsefulPoint=s_LE.ucUsefulPoint;
        }
}
```