第十一届“恩智浦”杯全国大学生

智能汽车竞赛

技 术 报 告

学　　校：集美大学诚毅学院

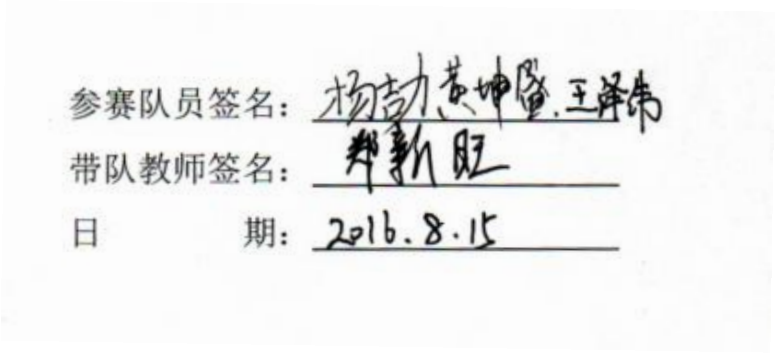
队伍名称：星轨队

参赛队员：杨劼 黄坤盛 王泽纬

带队教师：郑新旺 李颖

关于技术报告和研究论文使用授权的说明

本人完全了解第十一届“恩智浦”杯全国大学生智能汽车竞赛关保留、使用技术报告和研究论文的规定，即：参赛作品著作权归参赛者本人，比赛组委会和恩智浦半导体公司可以在相关主页上收录并公开参赛作品的设计方案、技术报告以及参赛模型车的视频、图像资料，并将相关内容编纂收录在组委会出版论文集中。



摘要

本文设计的智能车系统以 MK60N512ZVLQ10 微控制器为核心控制单元，通过摄像头 OV7725 检测赛道信息并用利用采集到的图像进行处理后算法计算偏差，用于赛道识别。利用基于位置式 PID 的模糊算法对小车进行舵机的横向控制。使用双向编码器对小车速度进行实时监控，使用基于位置式的自适应 PID 对小车进行电机纵向控制。本设计实现了对摄像头车运动速度和运动方向的闭环控制。为了提高模型车的速度和稳定性，使用了上位机、键盘模块，OLED 等调试工具，并进行了大量硬件与软件测试。实验结果表明，该系统设计方案确实可行。

关键字：MK60N512ZVLQ10　PID　模糊算法

目录

摘要 II
第一章 引言 1
1.1 比赛背景介绍 1
1.2 技术方案的实现 1
1.3 技术报告的结构安排 2
第二章 车模机械结构的调整 3
2.1 舵机的安装 3
2.2 前轮定位和调整 4
2.2.1 主销后倾角 4
2.2.2 主销内倾角 4
2.2.3 车轮外倾角 5
2.2.4 前轮前束 6
2.3 底盘高度的调整 6
2.4 编码器安装方式 7
2.5 前瞻的安装 7
2.5 防撞杆的安装 8
2.7 轮胎处理 9
2.8 差速调整 9
第三章 智能车硬件电路设计 10
3.1 单片机系统设计 10
3.2 电源模块设计 10
3.2.1 系统 5V 供电模块 11
3.2.2 舵机 5V 供电模块 11
3.2.3 摄像头和其他模块的 3.3V 供电模块 12
3.2.4 驱动电路 12V 供电模块 12
3.3 传感器方案 12
3.4 驱动电路设计 13
第四章 智能车系统软件设计 14
4.1 底层初始化 14
4.2 控制策略 15
4.2.1 PID 算法简介 15
4.2.2 控制器参数整定 17
4.2.3 模糊控制算法 18
4.2.4 模糊 PID 控制器建立 18
4.2.5 舵机的模糊 PD 控制 20
4.2.6 电机的自适应 PI 控制 20

4.2.7 路障识别 21
4.2.8 起跑线识别 21
第五章 系统开发及调试工具 22
5.1 开发调试工具 22
5.2 液晶屏、按键调试 22
第六章 车辆主要参数 24
第七章 总结 25
参 考 文 献 26
附录：源代码 27

第一章 引言

1.1 比赛背景介绍

全国大学生“恩智浦”杯智能汽车竞赛，等学校自动化专业教学指导分委员会主办，恩智浦半导体公司协办，以“立足培养，重在参与，鼓励探索，追求卓越”为指导思想，系统涵盖了控制、传感技术、机械、电子、电气、传感、计算机、自动化控制等多方面知识，每只参赛队伍要求制作一辆能够自主识别路径的智能车，在专门设计的跑道上自动识别道路行驶，最快跑完全程而没有冲出跑道的队伍为获胜者，且规则每年都有所修改，力求相对的公正公平，在一定程度上反映了高校学生科研水平。

第十一届“恩智浦”杯智能汽车竞赛摄像头组要求为单车竞技，不同于以往的规则，今年比赛赛道两侧加装路肩，更考验选手们对路肩算法的优化。在比赛当天，选手需在规定时间内发车，小车冲出起跑线后按规定路程行驶，期间不能不能冲出跑道，以冲过终点的时间作为最终成绩；参赛队伍可以由 3 位同学组成。车模必须使用 B 型车模。比赛跑道为表面白色光滑的PVC 耐磨塑胶地板，赛道边缘有宽度限制，赛道具体形状在比赛当天现场公布。各参赛队伍在严格遵守比赛规则的条件下，进行车模改装，并设计了主板、电机驱动等电路、并编写转向舵机控制等控制算法，以追求在最短的时间内跑完全程，并能检测赛道的起跑线，且能够在完成比赛之后自动停止在起跑线3m范围之内。

1.2 技术方案的实现

通过对比赛规则的详细解读之后，在硬件和结构方面，我们决定使用 B 车模作为我们的主体，其中系统主板采用飞思卡尔 32 位微控制器 MK60N512ZVLQ10 作为核心控制单元，OV7725 为道路信息采集传感器，速度传感器使用 512 线的双向增量型编码器，电机驱动采用 4 片 7843MOS 管，。在软件和算法方面，整个车体的控制系统采用闭环控制系统，利用摄像头采集到的赛道信息进行 PID 控制、舵机转向控制，以及整车赛道速度给定，利用 MCU 对速度传感器采集的脉冲信号进行处理，之后进行 PID 计算后自动调节 PWM 波

形输出，控制小车速度。

此外，我们还扩展了 OLED 显示屏和键盘模块作为人机操作界面，以便于智能小车的调试与相关参数调整。

车模的最终结构如图 1.1 所示。

图 1.1 车模的最终结构图

1.3 技术报告的结构安排

本技术报告是对参赛智能汽车制作技术方案、设计思路、制作过程、调试手段及其他相关技术研究的总结性报告。

报告共分为六个部分，其中第一章为引言部分，简单介绍比赛背景、总体设计方案及本文结构；第二章将详细介绍赛车的机械结构安装与调整过程，包括舵机、车体前轮倾角、重心的调整、传感器安装等；第三章重点介绍系统硬件电路的设计与实现过程；第四章是软件系统的设计与实现本章还将介绍转向及转速的 PID 控制策略。第五章详细介绍了赛车系统开发的调试工具。第六章是对模型车的主要技术参数汇总说明；最后第七章对本文进行总结概括。

第二章 车模机械结构的调整

2.1 舵机的安装

舵机安装将直接影响到前轮转向问题。如果舵机调整不到位，将很大程度上限制转向角度和转向灵敏度。车模原始设计方案是将舵机安装在了车模前部的中心位置，为了空间的合理利用，我们将舵机向前平移到了两个前轮之间，为小车主板留出了一定的空间。

我们参考往年的舵机安装方式发现舵机一般有立式安装和卧式安装两种，前期我们测试卧式安装和立式安装两者的区别，发现两者的区别在于，立式安装能够方便安装舵机臂，有利于调节赛车转向的中心值，卧式安装能够使重心降低。

通过查阅资料，阅读历年技术报告发现，长度较短的舵机连片优点是能够输出更大力矩，但是不足的是反应速度不够快，而对于长的舵机连片优点是反应速度较快，但是输出力矩不足且容易对舵机造成伤害。

最终，我们采用了立式舵机的安装方式，并选择了一套舵机连片(转向拉杆)，综合考虑了速度与力矩的关系，并根据模型车底盘的具体结构，简化了安装方式，实现了预期目标。

车模舵机最终安装图，如图 2.1 所示。

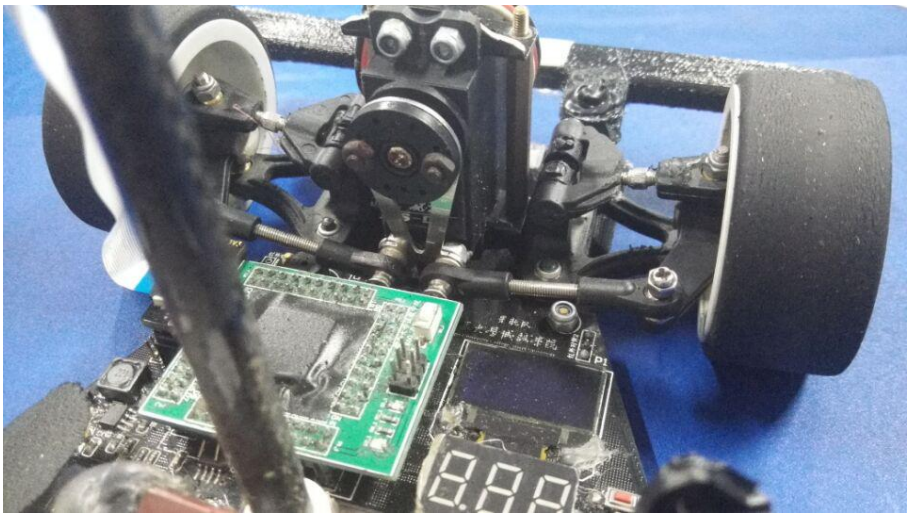


图 2.1　车模舵机最终安装图

2.2 前轮定位和调整

现代汽车在正常行驶过程中，为了使汽车直线行驶稳定，转向轻便，转向后能自动回正，减少轮胎和转向系零件的磨损等，在转向轮、转向节和前轴之间须形成一定的相对安装位置，叫车轮定位，其主要的参数有：主销后倾、主销内倾、车轮外倾和前束。

智能车竞赛模型车的四项参数理论上都可以进行调整，但是由于模型车加工和制造精度的问题，在通用的规律中还存在着一些偶然性，一切是实际调整的效果为准。

2.2.1 主销后倾角

主销后倾角是指在纵向平面内主销轴线与地面垂直线之间的夹角。它在车辆转弯时会产生与车轮偏转方向相反的回正力矩，使车轮自动恢复到原来的中间位置上。所以，主销后倾角越大，车速越高，前轮自动回正的能力就越强，但是过大的回正力矩会使车辆转向沉重。通常主销后倾角值设定在 1°到 3°。

模型车通过增减黄色垫片的数量来改变主销后倾角的，由于竞赛所用的转向舵机力矩不大，过大的主销后倾角会使转向变得沉重，转弯反应迟滞，所以设置为 0°，以便增加其转向的灵活性。

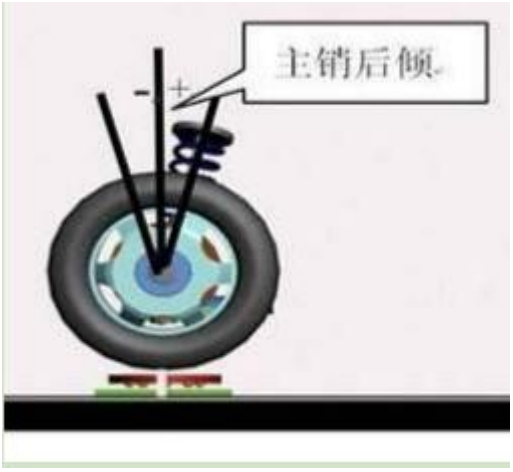


图 2.2　主销后倾

2.2.2 主销内倾角

主销内倾角是指横向平面内主销轴线与地面垂直线之间的夹角，它的作用

也是使前轮自动回正。角度越大前轮自动回正的作用就越强，但转向时也就越费力，轮胎磨损增大；反之，角度越小前轮自动回正的作用就越弱。通常汽车的主销内倾角不大于8°。

对于模型车，通过调整前桥螺杆的长度可以改变主销内倾角的大小，由于过大的内倾角也会增大转向阻力，增加轮胎磨损，所以在调整时可以近似调整为0°3°左右，不宜太大。

主销内倾和主销后倾都有使汽车转向自动回正，保持直线行驶的功能。不同之处是主销内倾的回正与车速无关，主销后倾的回正与车速有关，因此高速时主销后倾的回正作用大，低速时主销内倾的回正作用大。

2.2.3 车轮外倾角

前轮外倾角是指通过车轮中心的汽车横向平面与车轮平面的交线与地面垂线之间的夹角，对汽车的转向性能有直接影响，它的作用是提高前轮的转向安全性和转向操纵的轻便性。在汽车的横向平面内，轮胎呈“八”字型时称为“负外倾”，而呈现“V”字形张开时称为正外倾。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正外倾角度，一般这个角度约在1°左右，以减少承载轴承负荷，增加零件使用寿命，提高汽车的安全性能。

模型车提供了专门的外倾角调整配件，近似调节其外倾角。由于竞赛中模型主要用于竞速，所以要求尽量减轻重量，其底盘和前桥上承受的载荷不大，所以外倾角调整为0°即可，并且要与前轮前束匹配。

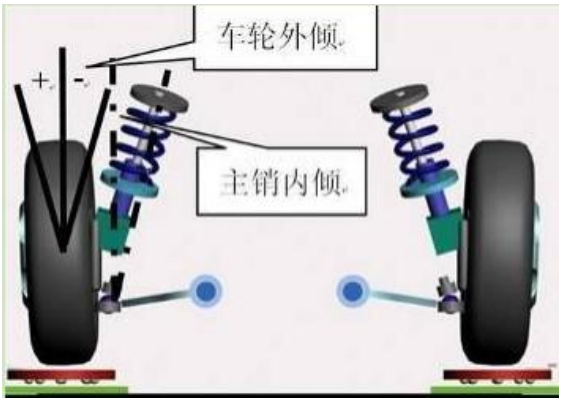


图2.3 主销内、外倾角

2.2.4 前轮前束

所谓前束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。像内八字那样前端小后端大的称为“前束”，反之则称为“后束”或“负前束”。在实际的汽车中，一般前束为 0-12mm。

在模型车中，前轮前束是通过调整伺服电机带动的左右横拉杆实现的。主销在垂直方向的位置确定后，改变左右横拉杆的长度即可以改变前轮前束的大小。在实际的调整过程中，我们发现较小的前束，约束 0-2mm 可以减小转向阻力，使模型车转向更为轻便，但实际效果不是十分明显。

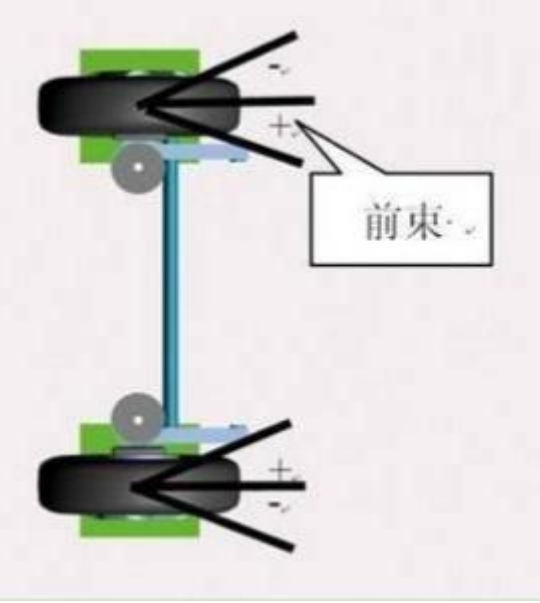


图 2.4　前轮前束图

2.3 底盘高度的调整

在保证顺利通过坡道、障碍的前提下，底盘尽量降低，我们将底盘进行紧固后，又将底盘的螺钉磨平，保证了底盘低平，从整体上降低模型车的重心，可使模型车转弯时更加稳定、高速。

由于过于靠前的重心将会增加舵机负担，而过后的重心又将会导致侧滑，经过多次试验，我们调整车模的机械结构，使车模的重心在合适的位置。

2.4 编码器安装方式

通常速度传感器可以选择光电码盘和光电编码器，其中光电码盘的重量轻，阻力小，精度较高，然而光电码盘长期暴露在外界容易受到外界干扰，例如光线或粉尘等的影响，导致计数不准确，而光电编码器就不存在此类问题，光电编码器精度高。因此为了精确的获得电机转速的返回值，我们采用 200 线的双向光电编码器，并使用配套的编码器安装支架，使编码器和齿轮紧密啮合，尽可能的使得传动齿轮轴保持平行，让传动部分更轻松、流畅，不存在过大噪音和丢数情况，并且使编码器安装位置尽量低，以降低车的重心。通过这样的安装方式实现了预期目标。

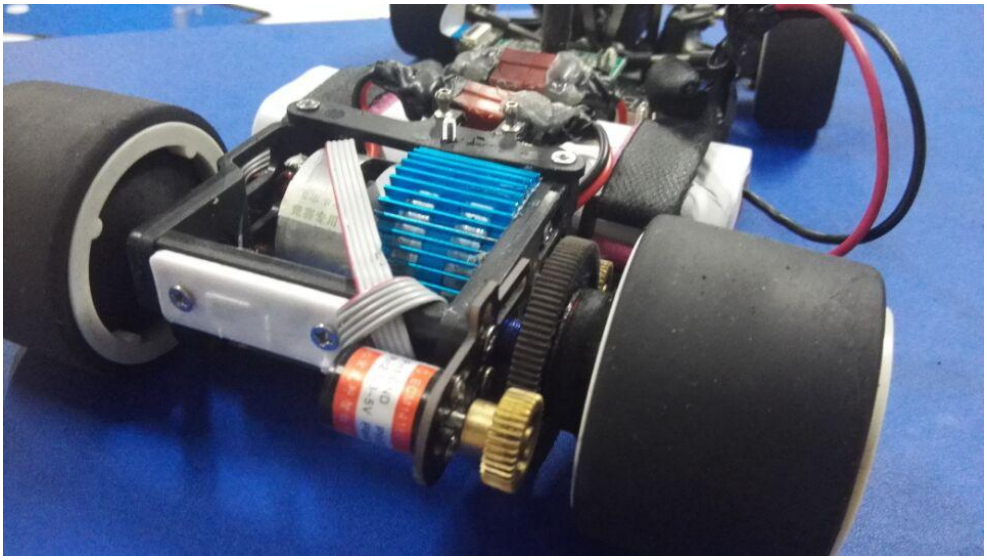


图 2.5 编码器的安装图

2.5 前瞻的安装

摄像头车前瞻的安装至关重要，前瞻的安装位置及高度会影响一辆车的性能好坏，前瞻距离越远就越能早的预测赛道位置并提前打角，但是过长的前瞻可能导致难于控制，严重情况下会导致小车串道，经过综合考虑，我们将前瞻长度设定在 150cm，兼顾了前瞻和稳定性。

由于传感器要延伸至小车上方，如果安装不牢固的话，将使得小车转向灵活度有很大影响。因此，在架设传感器的过程中，支架尽可能选择坚固且较轻的材料，并且要将传感器支架牢牢固定在车身上，否则小车行驶过车中传感器支架因不牢固而晃动将严重影响赛道信息的采集。

在这里我们使用 3D 打印技术打印的传感器支架底座（如图 2.6），能够牢牢将架牢牢固定在车身上。

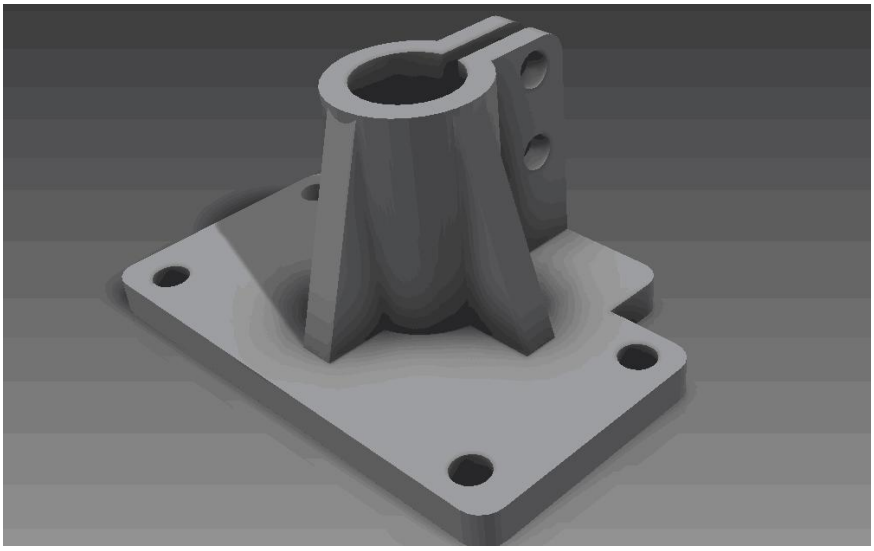


图 2.6 传感器支架底座

2.5 防撞杆的安装

B 车在高速运行时，因赛道误判等原因冲出跑道，撞击其他物体时，会因为撞击到前轮而导致舵机报废，因此我们采用 3D 打印技术，打印防撞杆，装在车头，防止因撞击而损坏车模部件，效果显著。

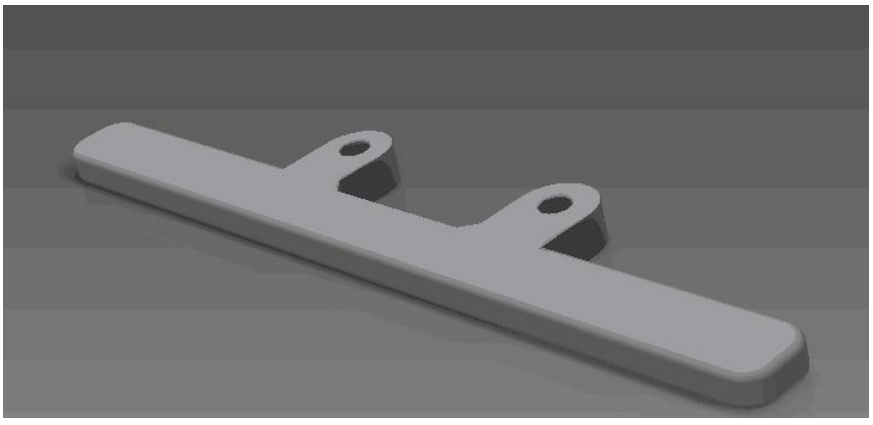


图 2.7 防撞杆

2.7 轮胎处理

轮胎的好坏对后期提速有着重要的作用，它的影响甚至强于算法本身和赛车的机械结构，但熟悉比赛规则的人都知道，组委会对赛车车模的轮子的检查格外严格。这主要是为了防止利用涂抹化学物质增大轮胎摩擦力而造成比赛不公平，如果轮胎检测不过关，将失去比赛资格。今年 B 车模的官方轮胎为带花纹的轮胎，规则允许使用旧车模的光轮胎，或者新车模的花纹轮胎。但不允许对旧的光轮胎进行花纹处理，同样不能对新的花纹轮胎进行除纹处理。

我们在不违反规则的前提下，对轮胎软化剂进行处理，前期效果显著，车速瞬间提升，但其明显的后果就在于，随着使用时间的增加，轮胎表面出现掉皮现象，且粘度增大，这已经违反比赛规则，最终我们只能放弃这一做法。

经过多次试验，我们发现不经任何处理的轮胎效果最佳。在正式比赛前几天，换新用新轮胎，轮胎以跑软且不掉碎渣为最佳状态，希望这一经验能使今后的参赛选手少走弯路。

2.8 差速调整

B 车模的差速为双差速结构，主要是对包括滚珠、差速片、橡胶垫圈，轴承等在内的零件以及差速器轴向的松紧程度进行调节。由于差速器是完全裸露在外面的，容易卷入脏东西而加速磨损，应该定期清理差速器的各个零件减小磨损并更换差速滚珠保证其圆度。差速轴向的松紧程度对于小车转向的灵敏性有很大影响，过松的差速可能使转向很灵敏出现过度转向，过紧的差速可能使转向很吃力出现转向不足。

评定差速器好坏的方法是给电机恒定转速使其带动后桥转动，用手握住单边轮另一侧轮能够反向转动且车轮的跳动度小，双手握住两轮中间差速齿轮不能够转动。

第三章 智能车硬件电路设计

3.1 单片机系统设计

单片机最小系统是智能车系统的核心部件,我们选择飞思卡尔公司提供的 Kinetis 系列的单片机 MK60N512ZVLQ10,由于该单片机具有丰富的资源，将其用于小车的控制核心，能完全满足小车的控制需求。

原理图如图 3.1 所示。

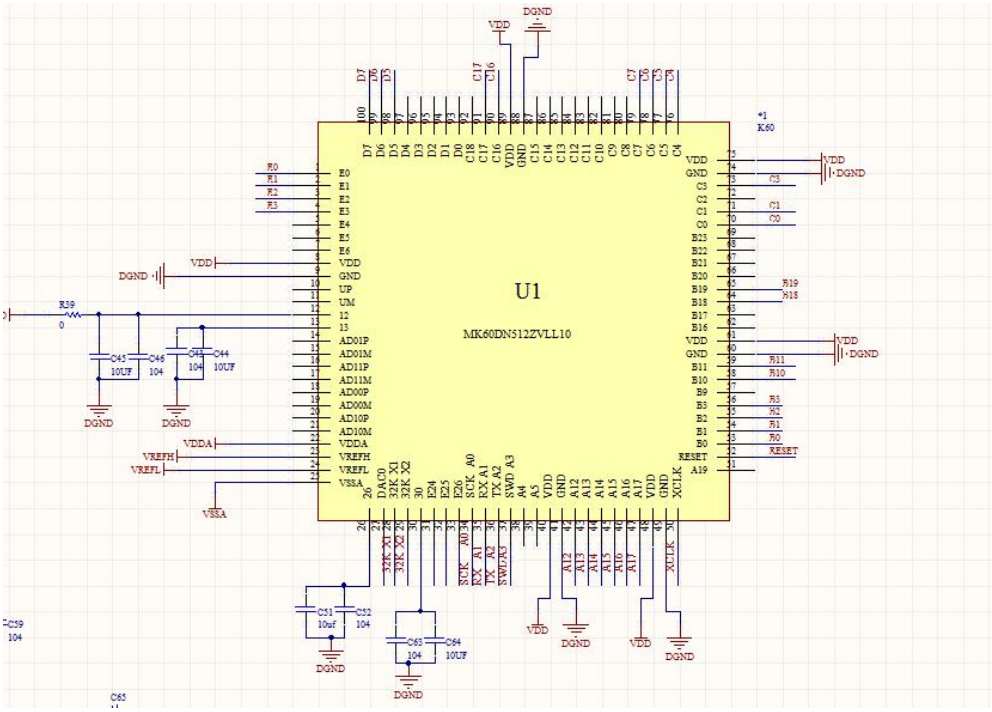


图 3.1　单片机系统原理图

3.2 电源模块设计

电源供电的稳定性直接关系到整个系统的稳定性。一个好的电源要考虑其转换效率，尽量降低噪声，防止干扰。竞赛规则规定，比赛使用智能车竞赛统一配发的标准车模用 7.2V 2000mAh Ni-cd 电池供电，因此设计一个合适的电源模块，对整车的稳定性起着决定性的作用。

由于电路中的不同电路模块所需要的工作电压和电流容量各不相同，因此电源模块应该包含多个稳压电路，将充电电池电压转换成各个模块所需要的电

压。主要包括以下不同的电源模块。

3.2.1 系统 5V 供电模块

此模块采用 LM2940 电压转换芯片将电池电压 7.2V 转换为 5V，为单片机、编码器、等模块提供稳定的 5V 电源。由于 LM2940 的稳压的线性度非常好，转换压差较小，发热现象并不明显，开关电源虽然有节约能源， 降低热损耗等优点，但由于其纹波较大，电压不稳而影响放大电路的正常使用，我们选用 LM2940 对其进行供电。

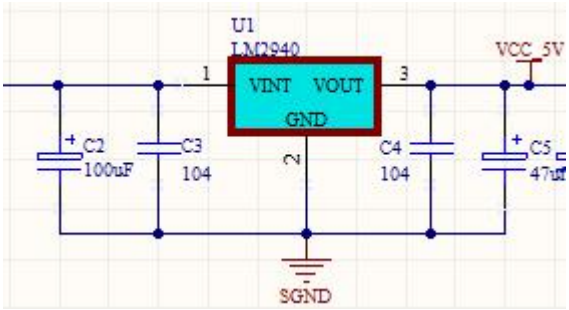


图 3.2　5V 稳压电路原理图

3.2.2 舵机 5V 供电模块

第十一届恩智浦摄像头组采用的是 B 车模，而 B 车模使用的伺服电机型号为 SD-5 该舵机基本参数如图 3.3 所示。为防烧坏舵机，舵机电源不能超过 5.5V，但电压太低会影响驱动效果。考虑到系统设计的简洁，因此我们在此系统中，直接使用 LM2940 设计产生 5V 的电源电路，但需注意的是，为防止干扰，我们的单独为舵机提供 5V 的供电。

S-D5 数码舵机参数

	数字舵机，塑胶齿，双含油轴承							4.5V			5.5V		
	尺寸 (mm)					重量		速度	扭力		速度	扭力	
	A	B	C	D	E	g	oz	sec/60°	kg-cm	oz-in	sec/60°	kg-cm	oz-in
	40.5	20.5	38	49.0	9.2	44	1.55	0.16	4.2		0.14	5	

此款舵机是特制的品种，工作电压只能在 5.5v 以下，有堵转保护功能，舵机在堵转后 3 秒后开始保护，降低电流，保护马达以及电板。

正常工作电流 200mA.

堵转电流 800mA

频率是 300HZ.

图 3.3　S-D5 数码舵机参数

3.2.3 摄像头和其他模块的 3.3V 供电模块

除了上述模块之外，本系统的摄像头、无线通信模块和显示模块 OLED 等都需使用 3.3V 供电。我们通过 LM2940 电压转换芯片将电池电压 7.2V 转换为 5V 之后，再利用 ASM1117 芯片将 5V 电压转换为 3.3V。

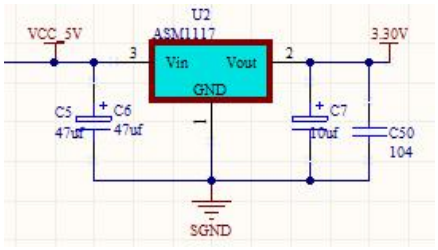


图 3.4　3.3V 稳压电路原理图

3.2.4 驱动电路 12V 供电模块

驱动电路采用12V的电源供电，由7.2V镍镉电池作为12V稳压电路的输入电压，由于电机驱动电路的电流较大，使用我们采用MC34063大电流升压变换器电路，得到12V的电压输出，12V稳压电路图如图3.5所示。

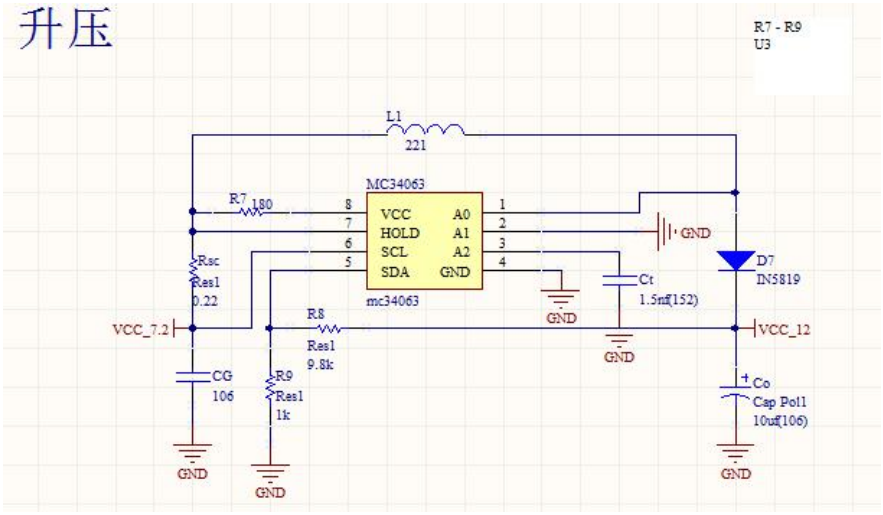


图 3.4　3.3V 稳压电路原理图

3.3 传感器方案

目前市场的模拟摄像头有两大种：CCD 和 CMOS。CCD 摄像头的优点是动态性能好，

即使车在高速行驶时也可得到较为清晰的图像，其缺点主要有两点：耗电量大、重量大、电路复杂。CMOS 摄像头恰好相反，动态性能较差，高速时容易出现图像模糊的现象，但体积小，耗电小、使用方便。

综合利弊，我们最终选择山外的 OV7725 鹰眼摄像头，OV7725 数字摄像头在硬件上做了二值化处理，所以阈值作为程序底层中一个十六进制的数值来体现，这里就不再赘述了。需要注意的是，如果想要获得效果明显清晰的图像，当环境改变时，阈值要做相应的调整来适应光线场地，这个是运行上层采线函数获得正确赛道信息的首要条件。

3.4 驱动电路设计

一个好的驱动模块电路对于竞速比赛的重要性是不言而喻的， 较好的加速与制动能力对小车平均速度的提高有很大帮助。初期我们选用电路比较简单的BTN7971作为电机的驱动芯片，小车能达到预期的要求。但随着小车速度提升，BTN7971发热问题愈发严重，通过多次电路改进均未取得显著效果，因此我们放弃选用BTN7971方案。

最终我们使用2片IR2104分别控制4片7843（N型MOSFET 管），搭建一个H桥电路，从而实现直流电机正反转，大大提高了电动机的工作转矩和转速。

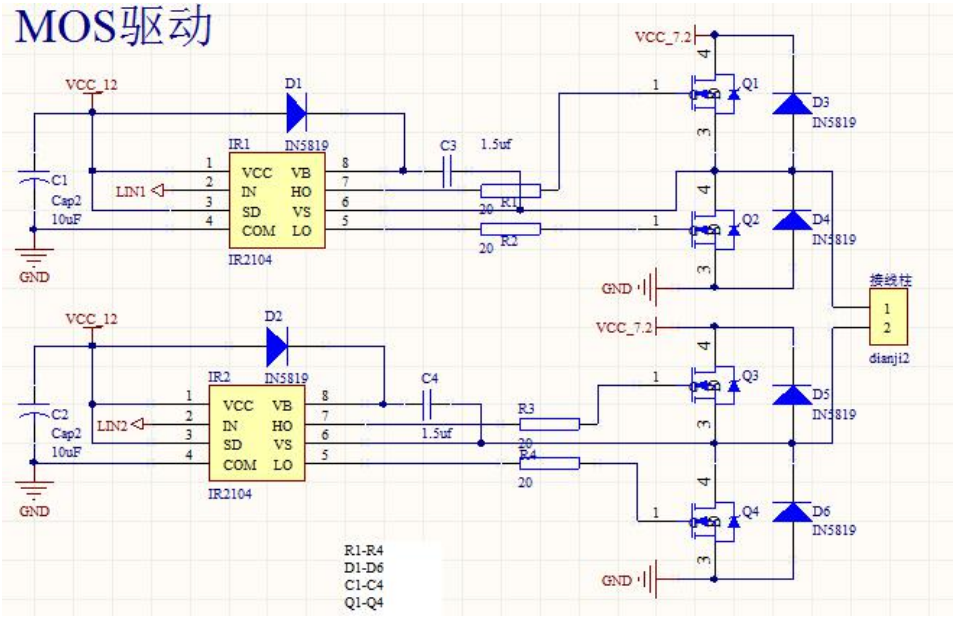


图 3.6 电机驱动原理图

第四章 智能车系统软件设计

本方案主要用到 MK60N512ZVLQ10 芯片中的 PIT 模块、ADC 模块、FTM 模块、PWM 模块、UART 模块、I/O 模块，Flash 模块。

4.1 底层初始化

```
void SystemInit(void)
{
    DisableInterrupts;          //关闭总中断
    Key_init();                 //按键初始化
    gpio_init(PORTC,14,1,1); //中断波形检测
    Flash_init();               //Flash初始化
    LCD_Init();                 //OLED初始化
    Pwm_init();                 //电机PWM初始化
    FTM_init();                 //正交解码初始化
    Steer_init();               //舵机初始化
    Adc_init(ADC1);                 //ADC1  初始化
    Uart_init();                //串口初始化
    Pit_init(PIT0);             //初始化  定时器0中断  1ms
    Pit_init(PIT1);                 //初始化  定时器1中断
    EnableInterrupts;               //开启总中断
}
```

其中，I/O模块主要是用来分配给按键模块、蜂鸣器、波形检测端口使用；Flash模块储存各类参数，开机自动读取；PIT模块用于提供定时中断，以便于对小车进行PID调节；ADC模块用于处理电磁信号的输入；FTM模块用于输出舵机、

电机所需要的PWM方波和采集编码器反馈回来的信号；UART模块用于传输数据到上位机。

4.2 控制策略

4.2.1 PID 算法简介

PID 控制器是控制系统中技术比较成熟， 而且应用最广泛的一种控制器。它具有不需要知道被控对象的数学模型，结构简单，参数容易调整，较强的灵活性和适应性的特点。因此在工业的各个领域中都有应用。在自动控制系统中，常用的控制器有比例—积分控制器（PI 控制器），比例—积分—微分控制器（PID 控制器）、分段式逼近控制器，较为新颖的有模糊控制器，神经元网络控制器等。

如原理框图 4.1 所示，按照偏差的比例、积分、微分进行控制的控制器称为 PID 控制器。其中 r(t)为系统给定值

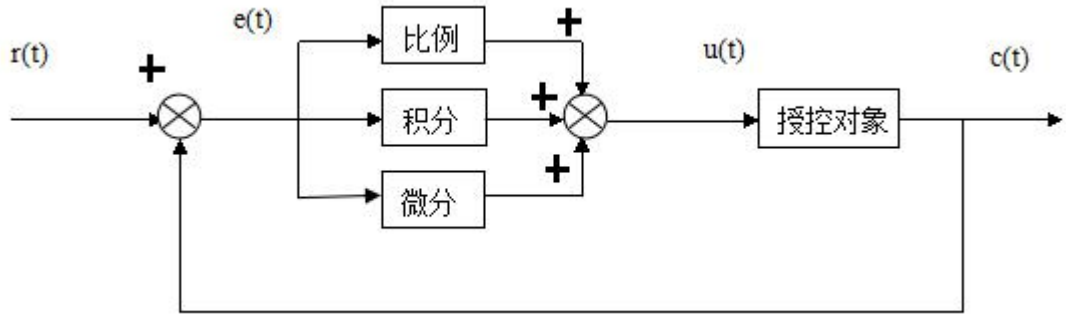


图 4.1 PID 控制器原理框图

如原理框图所示理想的 PID 控制器算式为：

$$u(t) = K_P[e(t) + \frac{1}{T_I}\int_0^t e(\tau)d\tau + T_D\frac{d(e(t))}{dt}] \qquad \text{(公式 4-1)}$$

式中：

u(t)——控制器(也称调节器)的输出；

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

Kp——控制器的比例放大系数；

Ti ——控制器的积分时间；

Td——控制器的微分时间。

由于单片机控制是一种离散的采样控制，在单片机系统中采用的是数字 PID 控制器。式（4-6）为模拟 PID 控制器表达式。通过模拟 PID 表达式中的积分、

$$u(k) = K_P e(k) + K_I \sum_{j=0}^{k} e(j) + K_D[e(k) - e(k-1)] \quad \text{(公式 4-2)}$$

$$K_{\mathrm{i}} = \frac{\mathrm{K_p} T}{T_i} \text{为积分系数}$$

$$K_{\mathrm{d}} = \frac{\mathrm{K_p} T_{\mathrm{d}}}{T_i} \text{为微分系数}$$

微分运算用数值计算方法来逼近，便可实现数字 PID 控制。只要采样周期 T 取得足够小，这种逼近方式就可以相当精确。

设 u(k)为第 k 次采样时刻控制器的输出值，可得离散的 PID 算式

其中(4-2)式中由单片机的输出 u(k)直接控制执行机构（如舵机），u(k)的值与执行机构的位置（舵机角度）一一对应，所以通常称式(4-2)为位置式 PID 控制算法。

位置式 PID 控制算法的缺点：当前采样时刻的输出与过去的各个状态有关，计算时要对 e(k)进行累加，运算量大；而且控制器的输出 u(k)对应的是执行机构

$$u(k-1) = K_P e(k-1) + K_I \sum_{j=0}^{k-1} e(j) + K_d[e(k-1) - e(k-2)] \quad \text{(公式 4-3)}$$

的实际位置，如果单片机出现故障，u(k)的大幅度变化会引起执行机构位置的大

$$\Delta u(k) = u(k) - u(k-1) \quad \text{(公式 4-4)}$$

$$= K_P[e(k) - e(k-1)] + K_I \times e(k) = K_D[e(k) - 2e(k-1) + e(k-2)]$$

幅度变化，这在生产过程中是不允许的。因而出现了增量式 PID 控制的控制算法。

所谓增量式 PID 是指数字控制器的输出只是控制量的的增量 U(k)。

由（公式 4-2）可得，第 k-1 时刻 PID 调节的表达式为：

将（公式 4-2）减去（公式 4-3），便可得到增量式 PID 控制算法：

PID 控制器各校正环节的作用如下：

比例调节（P）作用：是按比例反应系统的偏差，系统一旦出现了偏差，比例调节立即产生调节作用，以减少偏差。比例作用大，可以加快调节，减少误差，但是过大的比例，使系统的稳定性下降，甚至造成系统的不稳定。

积分调节（I）作用：是使系统消除稳态误差，提高无差度。因为有误差，积分调节就进行，直至无差，积分调节停止，积分调节输出一常值。积分作用的强弱取决与积分时间常数 Ti，Ti 越小，积分作用就越强。反之 Ti 大则积分作用弱，加入积分调节可使系统稳定性下降，动态响应变慢。积分作用常与另两种调节规律结合，组成 PI 调节器或 PID 调节器。

微分调节（D）作用：微分作用反映系统偏差信号的变化率，具有预见性，能预见偏差变化的趋势，因此能产生超前的控制作用，在偏差还没有形成之前，已被微分调节作用消除。因此，可以改善系统的动态性能。在微分时间选择合适情况下，可以减少超调，减少调节时间。微分作用对噪声干扰有放大作用，因此过强的加微分调节，对系统抗干扰不利。此外，微分反应的是变化率，而当输入没有变化时，微分作用输出为零。微分作用不能单独使用，需要与另外两种调节规律相结合，组成 PD 或 PID 控制器。

4.2.2 控制器参数整定

控制器参数整定：指决定调节器的比例系数 、积分时间 Ti 、微分时间 Td 和采样周期 Ts 的 Kp 如 PID 调节控制做电机速度控制具体数值整定的实质是通过改变调节器的参数，使其特性和过程特性相匹配，以改善系统的动态和静态指标，取得最佳的控制效果。

整定调节器参数的方法很多，归纳起来可分为两大类，即理论计算整定法和工程整定法。理论计算整定法有对数频率特性法和根轨迹法等；工程整定法有凑试法、临界比例法、经验法、衰减曲线法和响应曲线法等。工程整定法特点不需要事先知道过程的数学模型，直接在过程控制系统中进行现场整定方法简单、计算简便、易于掌握对于小车的调试我们最初采用无线模块传输数据到上位机，利用凑试法按照先比例 P 、再积分 I 、最后微分 D 的顺序。

置调节器积分时间 Ti＝∞，微分时间 Td＝0，在比例系数 Kp 按经验设置的初值条件下，将系统投入运行，由小到大整定比例系数 Kp 。求得满意的 1/4 衰减度过渡过程曲线。

引入积分作用（此时应将上述比例系数 KP 设置为 5/6 ） 。将 Ti 由大到

小进行整定。

若需引入微分作用时，则将 Td 按经验值或按 Td =（1/3～1/4） 设置，并由小到大加入。

4.2.3 模糊控制算法

一般控制系统包含了五个主要部分，即:定义变量、模糊化、知识库、逻辑判断及反模糊化，底下将就每一部分做简单的说明：

(1) 定义变量：也就是决定程序被观察的状况及考虑控制的动作，例如在一般控制问题上，输入变量有输出误差 E 与输出误差之变化率 CE，而控制变量则为下一个状态之输入 U。其中 E、CE、U 统称为模糊变量。

(2) 模糊化（fuzzify）：将输入值以适当的比例转换到论域的数值，利用口语化变量来描述测量物理量的过程，依适合的语言值（linguisitc value）求该值相对之隶属度，此口语化变量我们称之为模糊子集合（fuzzy subsets）。

(3) 知识库：包括数据库（data base）与规则库（rule base）两部分，其中数据库是提供处理模糊数据之相关定义；而规则库则藉由一群语言控制规则描述控制目标和策略。

(4) 逻辑判断：模仿人类下判断时的模糊概念，运用模糊逻辑和模糊推论法进行推论，而得到模糊控制讯号。此部分是模糊控制器的精髓所在。

(5) 解模糊化（defuzzify）：将推论所得到的模糊值转换为明确的控制讯号，做为系统的输入值。

模糊算法可以解决一些非线性问题，将赛道分为直线、入大小弯、出大小弯、蛇形弯道，对应的直线加速、入大弯减速转方向、入小弯制动转方向、出弯加速、蛇形弯道直接通过（若可以达到这种前瞻性）。要达到这种控制要通过实际检测，分析大量赛道磁场信息，找出它们的特征。

4.2.4 模糊 PID 控制器建立

模糊 PID 控制即是根据误差 error 与误差的变化率 e_error 依照模糊关系找到三个合适的 PID 参数。在运行中模糊控制规则在线调整三个参数来满足不同 error 与 e_error 的要求。

(1) 编码

模糊控制器有实际速度与期望速度的误差 error，前后两次 error 的变化率

e_error 两个输入量，输出量为 PID 控制器参数调整量ΔKp、ΔKi、ΔKd。控制器需要对着两个精确量进行编码使其模糊化。为后面模糊推理，模糊运算提供条件。根据实际情况设误差 error 基本论域为[-250,250]，划分 7 个等级即{-250,-186,-83,0,83,186,250}。模糊集取 7 个语言值分别为{NB,NM,NS,ZO,PS,PM,PB}；

变化率 e_error 基本论域为[-24,24]，划分 7 个等级即{-24,-16,-8,0,8,16,24}。模糊集取 7 个语言值分别为{NB,NM,NS,ZO,PS,PM,PB}；为了精确控制，输出语言变量ΔKp、ΔKd 取{NB,NM,NS,ZO,PS,PM,PB}7 个语言变量值。

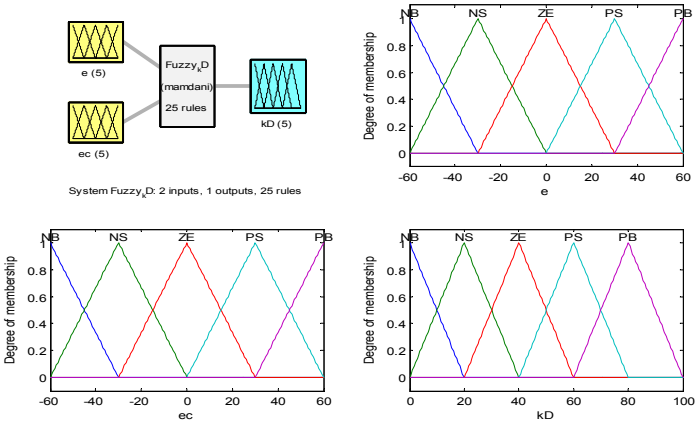


图 4.2 隶属度函数

(2) 模糊规则表

当| error |较大时，不论 e_error 的趋势如何，Kp 取较大的值使系统快速响应。防止| e_error |瞬间过大，Kd 取值小。为控制超调，Ki 取值小。

当| error |中等时，为控制超调和提高响应速度，则 Kp 减小，Kd 中等，Ki 值增大。

当| error |较小时，保证系统的稳定性，加大 Kp、Ki 的值。为了避免振荡 Kd 值根据| e_error |具体而定。

要想实现入弯快速打角，出弯完全后在快速回正，必须建立合适的规则表，根据对于 E 为正值时，入弯时 EC 为正，出弯时 EC 为负，而 E 为负值时，入弯时 EC 为负，出弯时 EC 为正，所以只要对四种情况下建立不同而又合适的隶属度规则便可以得到合适的速度.

我们经过分析大量赛道信息，并进行多次测试，最终整定出对应的规则表，如图 4.3 所示。

图 4.3 模糊规则表

(3) 解模糊

根据建立的模糊规则进行模糊推理后得到的是一个模糊量，不能直接用来控制被控对象。本文采用重心法来解模糊。其表达式为

$$Z_0 = \frac{\sum_{i=0}^{n} u_c(z_i) \bullet z_i}{\sum_{i=0}^{n} u_c(z_i)} \qquad \text{(公式 4-5)}$$

其中：Z_0 解模糊后的精确值；Z_i 模糊控制量论域内的值；$U_c(Z_i)$ 为 Z_i 的隶属度值。

4.2.5 舵机的模糊 PD 控制

基于这种模糊算法思想，我们根据输入偏差和其变化量的大小，通过建立适当的不同类型的模糊规则表，得到了入弯提前打角，出弯及时回整的效果。具体算法如下：

```
DJ_P = Fuzzy_Update(Kp,Error,Error_kpp);
DJ_D = Fuzzy_Update(Kd,Error,Error_kpp);
pservo=DJ_Kp*DJ_P *Error;
dservo=DJ_Kd*DJ_D*Error_Kpp;
DJ_set=DJ+pservo + dservo ;
```

其中 DJ_Kp,DJ_Kd 作为修正值，对数据进行修正。

4.2.6 电机的自适应 PI 控制

电机的闭环控制，我们采用基于位置式的自适应 PI 控制，在偏差大的情况

下，采用积分分离 PID，基本能实现直道高速，入弯迅速减速，出弯后再加速，解决了二次曲线出弯冲出赛道的问题，成功实现了小车的速度控制，使车具有良好的控速性能。

4.2.7 路障识别

通过识别出黑色路障的显著特征是对路障进行方向控制的核心。由路障的规格与摆放位置可得出其识别为跑道有一侧在一段时间内不出现跳变，而另一侧出现由白色跳变到黑色，且黑色段保持较多的行数，然后再跳变为白色。因而通过提取中线与设定的中线偏差得出相应的方向控制的量。再通过 PID 得出方向控制的输出量。

4.2.8 起跑线识别

根据比赛规则，起跑线只会出现在直道上，所以为了能够准确识别起跑线不至于在赛道弯道中识别错误，因此在程序中判断赛车在直道上行驶时才进行起跑线识别。起跑线的特点是两段横帖着的 10CM 的黑线，当赛车在直道上行驶时，能够看到整幅图像，因此两边沿黑线应该是最大行数的。又因为赛车高速运行中，起跑线采集可能只存在在一两行中，并且有可能不是一行中左右两段线都有，因此检测到一边也算作是检测到起跑线了。首先根据判定的三个条件中前两个直道，有效行数都满足的情况下，从图像的中间向两边寻找，如果某一行的黑点个数大于某一个值，小于某一个值，则认为检测到了起跑线。根据实际调试，能够比较准确的检测到起跑线。起跑线如图 4.4。

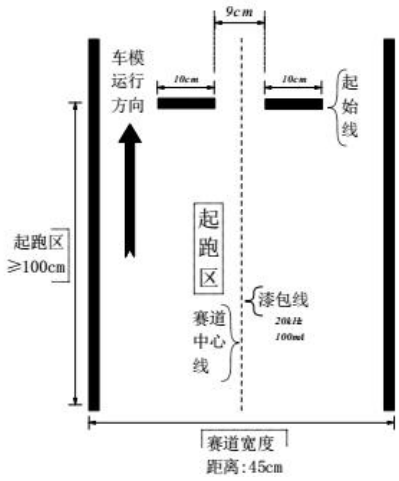


图 4.4 起跑线

第五章 系统开发及调试工具

5.1 开发调试工具

软件开发平台选用的是 Embedded Workbench for ARM，Embedded Workbench for ARM 是 IAR Systems 公司为 ARM 微处理器开发的一个集成开发环境(下面简称 IAR EWARM)。比较其他的 ARM 开发环境，IAR EWARM 具有入门容易、使用方便和代码紧凑等特点。它包含项目管理器、编辑器、C/C++编译器和 ARM 汇编器、连接器 XLINK 和支持 RTOS 的调试工具 C-SPY，为用户提供一个易学和具有最大 量代码继承能力的开发环境，以及对大多数和特殊目标的支持。嵌入式 IAR Embedded Workbench 有效提高用户的工作效率，通过 IAR 工具，可以大大节省软件调试时间。调试界面如图 5.1 所示：

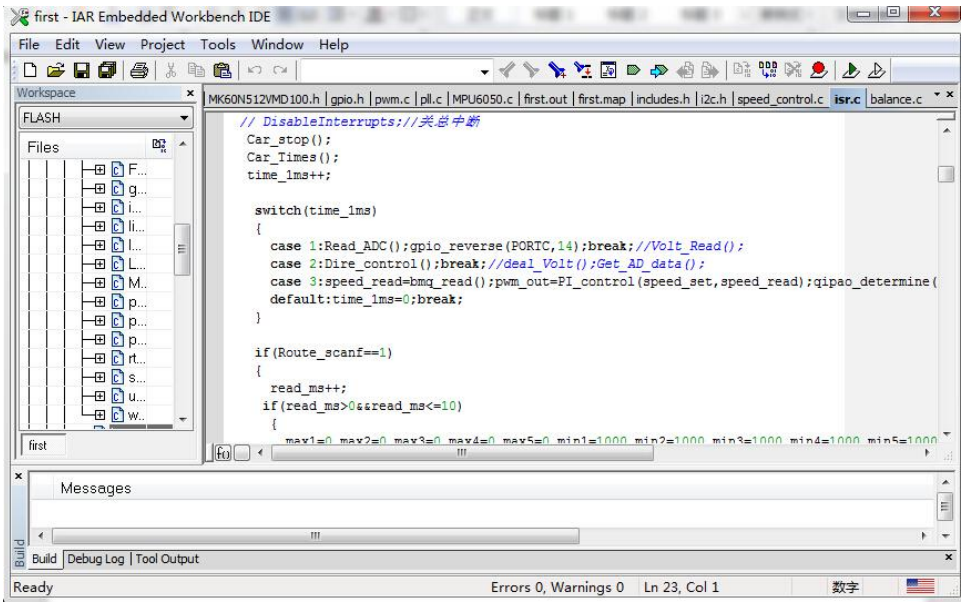


图 5.1 编译器调试界面

5.2 液晶屏、按键调试

日常调试过程中，我们需要不断调试，修正参数，因此我们选用了显示屏幕配合按键的调试方法。

在显示屏幕的选择上，我们选用了体积小的 OLED 液晶屏，它的尺寸为仅为 0.96 吋，具有普通 LCD 无法比拟的体积优势。

在按键的选择上，我们选用体积较小的贴片按键，并基于三行按键算法，仅需使用 4 个按键，便能满足我们的调试。

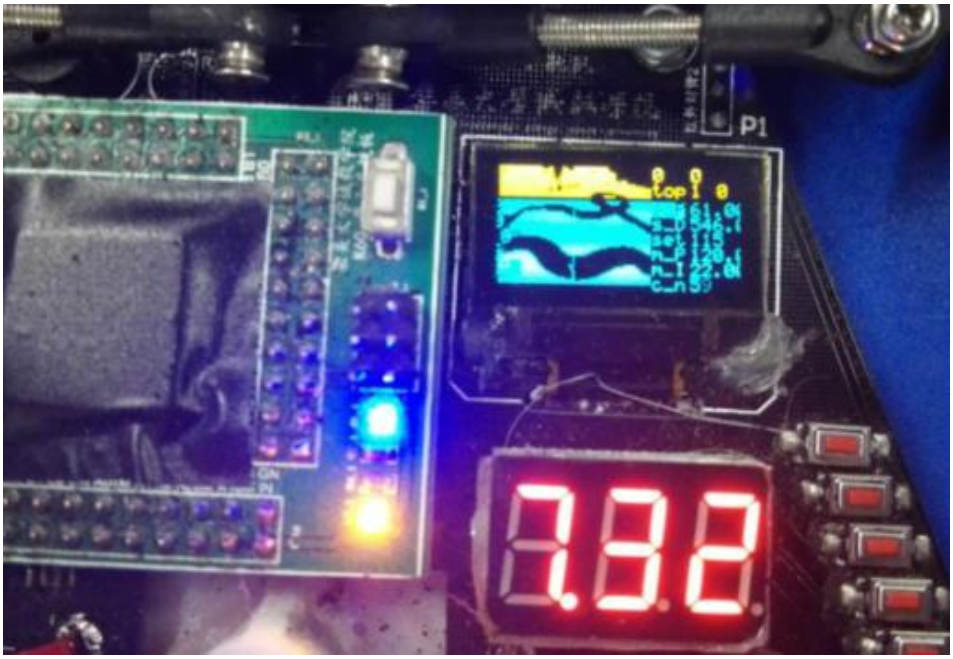


图 5.2 液晶屏、按键调试

第六章 车辆主要参数

车辆参数	参数
车模型号	B 车模
车模重量	1.2KG（含电池）
车模尺寸（长、宽、高）	300mm*190mm*350mm
车模传感器种类及个数	1 个摄像头、1 个编码器
电路电容总量（微法）	1900
赛道信息检测频率（次/秒）	1000
电路总功耗	16w

第七章 总结

从第十一届规则发布到赛区比赛结束，这九个月的时间里，我们小组成员从查找资料、设计机构、组装车模、编写程序一步一步改进，通过不懈努力，最后终于完成了最初目标，定下了现在这个设计方案。

这份技术报告，我们主要介绍了方案的整体思路，硬件结构上的搭建，电路的设计，以及智能车的控制算法。

在整个比赛的准备过程中，我们也遇到了很多困难。一开始选择组别的时候，我们选择双车组，并使用B车模作主体，但在调试的过程中，发现了B车灵活性不如C车，再加上大三学业繁重，无奈只能选摄像头组参赛，也算是没在一条路上走死。

从大一上学期的智能车展示会上，第一次接触飞思卡尔这比赛，便有了自己的一些想法，后来大一下学期有幸选被选上参加第九届华南赛，成为学长边上的小跟班，见证了学长们的努力的成果。直到大二这一年，和自己队友，从无到有，进了国赛拿到国二，再到这一年大三，潜心做车，最终取得华南赛第六的成绩，进入了决赛，智能车这三年，期间迷茫过，失望过，但却不曾放弃过，正如校训“诚以待人，毅以处事”做事有毅力，坚持不懈，不轻易放弃，方可见雨后彩虹。飞思卡尔一路走来，感慨万千，一辆车子，凝聚了我们智慧与汗水，见证了我们的努力与辛酸，每提升 0.1m/s 都是一个艰难的过程。虽说，在智能车比赛中带有一些运气成分，但我记得一个大神说过：我们之所以那么努力就是为了把运气成分降到最低。也正是这样的努力，让我们最后能够开心的笑。

在这几个月的备战过程中，场地和经费方面都得到了学院领导的大力支持，在此特别感谢一直支持和关注智能车比赛的学院领导以及各位指导老师、指导学长，同时也感谢比赛组委会能组织这样一项有意义的比赛。

参 考 文 献

(1) 邵贝贝．单片机嵌入式应用的在线开发方法 [M]．北京．清华大学出版社．2004.

(2) 余志生．汽车理论．北京．机械工业出版社．2012.

(3) 李仕伯,马旭,卓晴.基于磁场检测的寻线小车传感器布局研究[J].电子产品世界 2009.

(4) 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.

(5) 夏克俭．数据结构及算法 [M] ．北京：国防工业出版社， 2001.

(6) 贾翔羽,季厌庸,丁芳.前馈-改进 PID 算法在智能车控制上的应用

(7) 何宝祥,朱正伟,刘训飞,储开斌．模拟电路及其应用 [M] ．北京: 清华大学出版社，2008.

(8) 阎石．数字电子技术基础 [M] ．北京: 高等教育出版社，2000.

(9) 谭浩强著．C 程序设计．北京：清华大学出版社，2003．

(10) 李太福．基于在线参数自整定的模糊 PID 伺服控制系统[J] ．交流伺服系统，2005，4：203～215．

附录：源代码

```
void vcan_sendimg(uint8 *imgaddr, uint32 imgsize)
{
#define CMD_IMG         1
    uint8 cmdf[2] = {CMD_IMG, ~CMD_IMG};    //山外上位机 使用的命令
    uint8 cmdr[2] = {~CMD_IMG, CMD_IMG};    //山外上位机 使用的命令
    uart_putbuff(VCAN_PORT, cmdf, sizeof(cmdf));       //先发送命令
    uart_putbuff(VCAN_PORT, imgaddr, imgsize); //再发送图像
    uart_putbuff(VCAN_PORT, cmdr, sizeof(cmdr));       //先发送命令
}
void vcan_sendware(uint8 *wareaddr, uint32 waresize)
{
    #define CMD_WARE         3
    uint8 cmdf[2] = {CMD_WARE, ~CMD_WARE}; //串口调试使用的命令
uint8 cmdr[2] = {~CMD_WARE, CMD_WARE};//串口调试 使用的命令
uart_putbuff(VCAN_PORT, cmdf, sizeof(cmdf));       //先发送命令
uart_putbuff(VCAN_PORT, wareaddr, waresize); //再发送图像
uart_putbuff(VCAN_PORT, cmdr, sizeof(cmdr));       //先发送命令
}
/*****************************
Fuction: Caculate Fuzzy Value
Par.     : FuzzyStruct* F_S
             float ek
             float ekc
Return : Fuzzy Value
*****************************/
float Fuzzy_Update(FuzzyStruct* F_S,float ek,float ekc)
{
  float value=0;
```

```
    F_S->fe=ek;
    F_S->fec=ekc;
    value=FuzzyCtrl(F_S);
    return value;
}//...end Fuzzy_Update();
float FuzzyCtrl(FuzzyStruct* Fuzzy_S)
{
    float   eFuzzy[2]   ={0.0, 0.0};
    float   ecFuzzy[2] ={0.0, 0.0};
    float   U1Fuzzy[7] ={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0};
    int   num=0,pe=0,pec=0;
    float kp=0.0;
    int rank;
    int i=0;
    rank=Fuzzy_S->Rank;

    switch(rank)
    {
    case 7:
        {
            /*-----误差隶属函数描述-----*/
            if(Fuzzy_S->fe < Fuzzy_S->eRule[0])   // |x_x_x_x_x_x_x_
            {
                eFuzzy[0] =1.0;
                pe= 0;
            }
            else if(Fuzzy_S->fe < Fuzzy_S->eRule[1]) // _x|x_x_x_x_x_x_
            {
                eFuzzy[0]                                                    =
(Fuzzy_S->eRule[1]-Fuzzy_S->fe)/(Fuzzy_S->eRule[1]-Fuzzy_S->eRule[0]);
                pe = 0;
```

```
                }
                else if(Fuzzy_S->fe < Fuzzy_S->eRule[2]) // _x_x|x_x_x_x_x_
                {
                    eFuzzy[0]                =                (Fuzzy_S->eRule[2]
-Fuzzy_S->fe)/(Fuzzy_S->eRule[2]-Fuzzy_S->eRule[1]);
                    pe =1;
                }
                else if(Fuzzy_S->fe < Fuzzy_S->eRule[3]) // _x_x_x|x_x_x_x_
                {
                    eFuzzy[0]                =                (Fuzzy_S->eRule[3]
-Fuzzy_S->fe)/(Fuzzy_S->eRule[3]-Fuzzy_S->eRule[2]);
                    pe =2;
                }
                else if(Fuzzy_S->fe<Fuzzy_S->eRule[4])    // _x_x_x_x|x_x_x_
                {
                    eFuzzy[0]                                                =
(Fuzzy_S->eRule[4]-Fuzzy_S->fe)/(Fuzzy_S->eRule[4]-Fuzzy_S->eRule[3]);
                    pe=3;
                }
                else if(Fuzzy_S->fe<Fuzzy_S->eRule[5])   // _x_x_x_x_x|x_x_
                {
                    eFuzzy[0]                                                =
(Fuzzy_S->eRule[5]-Fuzzy_S->fe)/(Fuzzy_S->eRule[5]-Fuzzy_S->eRule[4]);
                    pe=4;
                }
                else if(Fuzzy_S->fe<Fuzzy_S->eRule[6])   // _x_x_x_x_x_x|x_
                {
                    eFuzzy[0]                                                =
(Fuzzy_S->eRule[6]-Fuzzy_S->fe)/(Fuzzy_S->eRule[6]-Fuzzy_S->eRule[5]);
                    pe=5;
                }
```

```
        else                                    // _x_x_x_x_x_x_x|
        {
            eFuzzy[0] =1.0;
            pe=6;
        }
        eFuzzy[1] = 1.0 - eFuzzy[0];

        /*-----误差变化隶属函数描述-----*/
        if(Fuzzy_S->fec < Fuzzy_S->ecRule[0])
        {
            ecFuzzy[0] =1.0;
            pec = 0;
        }
        else if(Fuzzy_S->fec < Fuzzy_S->ecRule[1])
        {
            ecFuzzy[0]          =          (Fuzzy_S->ecRule[1]          -
Fuzzy_S->fec)/(Fuzzy_S->ecRule[1]-Fuzzy_S->ecRule[0]);
            pec = 0 ;
        }
        else if(Fuzzy_S->fec < Fuzzy_S->ecRule[2])
        {
            ecFuzzy[0]          =          (Fuzzy_S->ecRule[2]          -
Fuzzy_S->fec)/(Fuzzy_S->ecRule[2]-Fuzzy_S->ecRule[1]);
            pec = 1;
        }
        else if(Fuzzy_S->fec < Fuzzy_S->ecRule[3])
        {

            ecFuzzy[0]          =          (Fuzzy_S->ecRule[3]          -
Fuzzy_S->fec)/(Fuzzy_S->ecRule[3]-Fuzzy_S->ecRule[2]);
            pec = 2 ;
```

```
                }
                else if(Fuzzy_S->fec < Fuzzy_S->ecRule[4])
                {                    ecFuzzy[0]    =    (Fuzzy_S->ecRule[4]    -
Fuzzy_S->fec)/(Fuzzy_S->ecRule[4]-Fuzzy_S->ecRule[3]);
                    pec=3;
                }
                else if(Fuzzy_S->fec < Fuzzy_S->ecRule[5])
                {
                    eFuzzy[0]                                                    =
(Fuzzy_S->ecRule[5]-Fuzzy_S->fec)/(Fuzzy_S->ecRule[5]-Fuzzy_S->ecRule[4]);
                    pec=4;
                }
                else if(Fuzzy_S->fec<Fuzzy_S->ecRule[6])
                {
                    eFuzzy[0]                                                    =
(Fuzzy_S->ecRule[6]-Fuzzy_S->fec)/(Fuzzy_S->ecRule[6]-Fuzzy_S->ecRule[5]);
                    pec=5;
                }
                else
                {
                    ecFuzzy[0] =1.0;
                    pec=6;
                }

                ecFuzzy[1] = 1.0 - ecFuzzy[0];
                break;
            }//...end case 7
        case 5:
            {
                                /*-----误差隶属函数描述-----*/
                if(Fuzzy_S->fe < Fuzzy_S->eRule[0])             // |x_x_x_x_x_
```

```
                {
                    eFuzzy[0] =1.0;
                    pe= 0;
                }
                else if(Fuzzy_S->fe < Fuzzy_S->eRule[1]) // _x|x_x_x_x_
                {
                    eFuzzy[0]                                                    =
(Fuzzy_S->eRule[1]-Fuzzy_S->fe)/(Fuzzy_S->eRule[1]-Fuzzy_S->eRule[0]);
                    pe = 0;
                }
                else if(Fuzzy_S->fe < Fuzzy_S->eRule[2])    // _x_x|x_x_x_
                {
                    eFuzzy[0]                =                (Fuzzy_S->eRule[2]
-Fuzzy_S->fe)/(Fuzzy_S->eRule[2]-Fuzzy_S->eRule[1]);
                    pe =1;
                }
                else if(Fuzzy_S->fe < Fuzzy_S->eRule[3])    // _x_x_x|x_x_
                {
                    eFuzzy[0]                =                (Fuzzy_S->eRule[3]
-Fuzzy_S->fe)/(Fuzzy_S->eRule[3]-Fuzzy_S->eRule[2]);
                    pe =2;
                }
                else if(Fuzzy_S->fe<Fuzzy_S->eRule[4])     // _x_x_x_x|x_
                {                                                eFuzzy[0]                =
(Fuzzy_S->eRule[4]-Fuzzy_S->fe)/(Fuzzy_S->eRule[4]-Fuzzy_S->eRule[3]);
                    pe=3;
                }
                else                                // _x_x_x_x_x|
                {
                    eFuzzy[0] =1.0;
                    pe =4;
```

```
        }
        eFuzzy[1] = 1.0 - eFuzzy[0];

        /*-----误差变化隶属函数描述-----*/
        if(Fuzzy_S->fec < Fuzzy_S->ecRule[0])
        {
            ecFuzzy[0] =1.0;
            pec = 0;
        }
        else if(Fuzzy_S->fec < Fuzzy_S->ecRule[1])
        {
            ecFuzzy[0] = (Fuzzy_S->ecRule[1] -
    Fuzzy_S->fec)/(Fuzzy_S->ecRule[1]-Fuzzy_S->ecRule[0]);
            pec = 0 ;
        }
        else if(Fuzzy_S->fec < Fuzzy_S->ecRule[2])
        {
            ecFuzzy[0]          =          (Fuzzy_S->ecRule[2]          -
Fuzzy_S->fec)/(Fuzzy_S->ecRule[2]-Fuzzy_S->ecRule[1]);
            pec = 1;
        }
        else if(Fuzzy_S->fec < Fuzzy_S->ecRule[3])
        {
            ecFuzzy[0]          =          (Fuzzy_S->ecRule[3]          -
Fuzzy_S->fec)/(Fuzzy_S->ecRule[3]-Fuzzy_S->ecRule[2]);
            pec = 2 ;
        }
        else if(Fuzzy_S->fec < Fuzzy_S->ecRule[4])
        {                   ecFuzzy[0]      =      (Fuzzy_S->ecRule[4]      -
Fuzzy_S->fec)/(Fuzzy_S->ecRule[4]-Fuzzy_S->ecRule[3]);
            pec = 3;
```

```
                }
                else
                {
                    ecFuzzy[0] =1.0;
                    pec = 4;
                }

                ecFuzzy[1] = 1.0 - ecFuzzy[0];
                break;
            }//...end case 5
        case 3:
            {
                                /*-----误差隶属函数描述-----*/
                if(Fuzzy_S->fe < Fuzzy_S->eRule[0])             // |x_x_x_
                {
                    eFuzzy[0] =1.0;
                    pe= 0;
                }
                else if(Fuzzy_S->fe < Fuzzy_S->eRule[1])        // _x|x_x_
                {
                    eFuzzy[0] =
    (Fuzzy_S->eRule[1]-Fuzzy_S->fe)/(Fuzzy_S->eRule[1]-Fuzzy_S->eRule[0]);
                    pe = 0;
                }
                else if(Fuzzy_S->fe < Fuzzy_S->eRule[2])        // _x_x|x_
                {
                    eFuzzy[0]                   =                   (Fuzzy_S->eRule[2]
-Fuzzy_S->fe)/(Fuzzy_S->eRule[2]-Fuzzy_S->eRule[1]);
                    pe =1;
                }
                else                                // _x_x_x|
```

```
{
    eFuzzy[0] =1.0;
    pe =2;
}
eFuzzy[1] = 1.0 - eFuzzy[0];

/*-----误差变化隶属函数描述-----*/
if(Fuzzy_S->fec < Fuzzy_S->ecRule[0])
{
    ecFuzzy[0] =1.0;
    pec = 0;
}
else if(Fuzzy_S->fec < Fuzzy_S->ecRule[1])
{
    ecFuzzy[0] = (Fuzzy_S->ecRule[1] - Fuzzy_S->fec)/(Fuzzy_S->ecRule[1]-Fuzzy_S->ecRule[0]);
    pec = 0 ;
}
else if(Fuzzy_S->fec < Fuzzy_S->ecRule[2])
{
    ecFuzzy[0] = (Fuzzy_S->ecRule[2] - Fuzzy_S->fec)/(Fuzzy_S->ecRule[2]-Fuzzy_S->ecRule[1]);
    pec = 1;
}
else
{
    ecFuzzy[0] =1.0;
    pec = 2;
}
ecFuzzy[1] = 1.0 - ecFuzzy[0];
break;
```

```
        }//...end case 3
    default: break;
    }//...end switch
    /*查询模糊规则表*/
    if(pe<(rank-1) && pec<(rank-1))              // e 和 e'都没有达到边缘
    {
        num =Fuzzy_S->rule[pec][pe];
        U1Fuzzy[num] += eFuzzy[0]*ecFuzzy[0];
        num =Fuzzy_S->rule[pec][pe+1];
        U1Fuzzy[num] += eFuzzy[1]*ecFuzzy[0];
        num =Fuzzy_S->rule[pec+1][pe];
        U1Fuzzy[num] += eFuzzy[0]*ecFuzzy[1];
        num =Fuzzy_S->rule[pec+1][pe+1];
        U1Fuzzy[num] += eFuzzy[1]*ecFuzzy[1];
    }
    else if(pe==(rank-1) && pec<(rank-1))    // e 达到边缘
    {
        num =Fuzzy_S->rule[pec][pe];
        U1Fuzzy[num] += eFuzzy[0]*ecFuzzy[0];
        num =Fuzzy_S->rule[pec+1][pe];
        U1Fuzzy[num] += eFuzzy[0]*ecFuzzy[1];
    }
    else if(pe<(rank-1) && pec==(rank-1))    // e'达到边缘
    {
        num =Fuzzy_S->rule[pec][pe];
        U1Fuzzy[num] += eFuzzy[0]*ecFuzzy[0];
        num =Fuzzy_S->rule[pec][pe+1];
        U1Fuzzy[num] += eFuzzy[1]*ecFuzzy[0];
    }
    else                                     // e 和 e'同时达到边缘
    {
```

```
            num =Fuzzy_S->rule[pec][pe];
            U1Fuzzy[num] += eFuzzy[0]*ecFuzzy[0];
        }
        /*面积中心法解模糊*/
        for(i=0;i<rank;i++)
            kp+=U1Fuzzy[i]*Fuzzy_S->U1Rule[i];
         return(kp);
}
```