天津中德应用技术大学

Tianjin Sino-German University of Applied Sciences

本科生毕业设计

面向智能车竞赛的四轮竞速车设计与调试

Design and Debugging of Four-Wheel Racing Cars for Intelligent Car Competition

姓　　名	郅淇贤
学　　院	汽车与轨道交通学院
专　　业	汽车服务工程
指导老师	温国强
职　　称	副教授
完成时间	**2023-5-20**

天津中德应用技术大学

本科生毕业设计（论文）的声明

本人郑重声明：所呈交的毕业设计（论文），是本人在指导教师指导下，进行研究工作所取得的成果。除文中已经注明引用的内容外，本毕业设计（论文）的研究成果不包含任何他人创作的、已公开发表或没有公开发表的作品内容。对本设计（论文）所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。本毕业设计（论文）原创性声明的法律责任由本人承担。

毕业设计（论文）作者签名：邳淇贤

年　　月　　日

本人声明：该毕业设计（论文）是本人指导学生完成的研究成果，已经审阅过设计（论文）的全部内容，并能够保证题目、关键词、摘要部分中英文内容的一致性和准确性。

毕业设计（论文）指导教师签名：

年　　月　　日

摘　要

智能汽车是当前智能化技术和新能源技术的重要应用方向，是推动汽车产业技术升级和城市交通管理创新的重要手段。本文是以第 17 届全国大学生智能车竞赛规则为基础，以大赛组委会指定的 C 型车模为平台所采用的智能车系统方案。该方案由主控模块、电机驱动模块、舵机转向模块和传感器模块组成，其中采用了 TC 单片机作为系统控制处理器。该智能车系统具有自动停车和准确识别赛道特殊元素等功能，这些功能的实现离不开各个模块之间的协调配合。主控模块是智能车系统的核心，通过对传感器模块获取的赛道信息进行数据处理，实现智能巡线和避障功能。电机驱动模块是智能车系统的重要组成部分，通过对电机进行驱动和控制，实现智能车的前进、左转和右转等操作。舵机转向模块则负责控制智能车的转向角度，保证智能车能够沿着赛道正确行驶。传感器模块是智能车系统的“眼睛”，通过对赛道信息进行采集和处理，实现智能车的巡线和避障功能。该模块采用摄像头传感器和编码器，能够对赛道上的障碍物、黑线和白线等元素进行准确识别和判断。而编码器可以计算车轮的转速并输出反馈信号，为速度控制提供参考。具体来说，摄像头被用作循迹传感器，编码器作为速度反馈传感器。同时，采用增量式 PID 算法完成对车速的闭环控制，PWM 控制驱动电路以调整电机功率，从而实现对车速和方向的精准控制。最终，智能车可以在赛道上完成比赛任务。本文的研究成果对智能车技术的发展和应用具有参考意义。

关键词：智能车；摄像头；增量式 PID 算法；电感电压控制

ABSTRACT

Intelligent vehicles are an important application direction of current intelligent technology and new energy technology, which serve as a crucial means to promote technological upgrading of the automobile industry and innovation in urban traffic management. This article is based on the rules of the 17th National University Intelligent Vehicle Competition and introduces the intelligent vehicle system scheme used by taking the C-type car model designated by the competition organizing committee as a platform. The scheme consists of a main control module, a motor drive module, a servo steering module, and a sensor module, with the TC MCU used as the system control processor. The intelligent vehicle system can perform functions such as automatic parking and accurate recognition of special elements on the track, which require coordination and cooperation between various modules. The main control module is the core of the intelligent vehicle system, processing data obtained from the sensor module to achieve intelligent line following and obstacle avoidance. The motor drive module is an essential component of the intelligent vehicle system, driving and controlling the motor to perform operations such as moving forward or backward, turning left or right. The servo steering module controls the vehicle's turning angle, ensuring that the vehicle follows the track correctly. The sensor module serves as the "eyes" of the intelligent vehicle system, collecting and processing track information to enable line following and obstacle avoidance, and uses a camera sensor to accurately recognize and judge obstacles, black lines, and white lines on the track. The encoder can calculate the rotation speed of the wheels and output feedback signals to provide reference for speed control. Specifically, the camera is used as the tracking sensor, the encoder serves as the speed feedback sensor, and an incremental PID algorithm is adopted to perform closed-loop control of the vehicle's speed. PWM controls the drive circuit to adjust motor power and allow precise control over the vehicle's speed and direction. Finally, the intelligent vehicle can complete competition tasks on the track. The research results detailed in this paper have reference significance for the development and application of intelligent vehicle technology.

Keywords: Intelligent Car；Camera,；Incremental PID Algorithm；Inductance Voltage Control

目 录

第一章　绪论

1.1 背景介绍

1.1.1 全国大学生智能汽车竞赛背景介绍

全国大学生智能汽车竞赛受教育部高等教育司委托，由教育部高等自动化专业教学指导委员会（以下简称自动化教指委）主办并负责。作为以智能汽车为研究对象的创意性科技竞赛，其不仅是面向全国大学生的一种具有探索性工程实践活动，也是教育部倡导的大学生科技竞赛之一。该竞赛已发展成全国 30 个省市自治区近 300 所高校广泛参与的全国大学生智能汽车竞赛。2008 年起被教育部批准列入国家教学质量与教学改革工程资助项目中科技人文竞赛之一（教高函[2007]30 号文）[1]。

全国大学生智能车竞赛由竞赛秘书处设计、规范标准硬软件技术平台，竞赛 过程包括理论设计、实际制作、整车调试、现场比赛等环节，也正体现了该竞赛“立足培养、重在参与、鼓励探索、追求卓越”的指导思想。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。其规则透明，评价标准客观，坚持公开、公平、公正的原则，保证竞赛向健康、普及、持续的方向发展。

为了实现竞赛的"立足培养、重在参与、鼓励探索、追求卓越"的指导思想，竞赛内容设置需要能够面向高校学生和教学内容，同时又能够兼顾当今时代科技发展的新趋势。比赛形式包括有竞速比赛与创意比赛两大类。竞速比赛中包含不同的组别，难度适合高校不同年级学生参赛。在竞速赛基础上，适当增加挑战性，形成创意比赛的内容，适合部分有条件、能力强的本科、专科生和研究生参加创意比赛。

为了兼顾现在比赛规模的要求，同时避免同组别内出现克隆车的情况，便于参赛学校在有限的场地内使用兼容的赛道完成比赛准备，竞速比赛将按九个组别(四轮竞速组包括四轮电磁组和四轮摄像头组、多车编队组、平衡单车组、无线充电组、平衡信标组、智能视觉组、极速越野组、完全模型组)进行设置。

1.1.2 智能车竞赛规则介绍

本届大赛要求基础四轮组使用 C 型车模，车模运行方向不限，若安装摄像头传感器，摄像头镜片中心的高度距离地面不超过 15cm。车模必须采用 TC 系列单片机作为唯一车模微控制器。选手制作的车模完成从车库出发在赛道上往返一周后，然后在返回车库。比赛时间从车模驶出车库到重新回到车库为止，也就是说车模需要能够识别赛道上的起跑线标志完成车模的出发和停止控制。

1.2 整车设计总述

智能车的设计是一项非常有挑战性的任务，需要各个方面的技能和知识。本文将深入介绍智能车的设计和调试过程，包括机械设计、电路设计和软件设计等方面。每个方面都是制作智能车过程中非常重要的一环，每一个步骤都需要认真谨慎地完成[2]。

(1)机械设计部分分为制作和调整两部分。在制作部分，需要进行车模的基础组装和对缺失部件的设计。例如，传感器支架和电路板固定等。这些都是非常重要的组成部分，需要精心设计和制作。在调整部分，需要在竞赛规则允许范围内针对车模本身机械部分进行改装和调整，以提高其运动性能，适应高速行驶和快速控制。例如，改装舵机的固定结构和调整车轮定位等可以有效提高智能车的性能。

(2)电路设计部分包括单片机最小系统、电源模块、传感器模块和驱动模块等模块。这些模块遵循紧凑、易于拆换、稳定可靠的总体设计原则，但是不同模块有着不同的设计要求。例如，单片机最小系统需要满足稳定可靠、易于维护等要求，电源模块需要满足高效稳定、低功耗等要求，传感器模块需要满足高精度、高灵敏度等要求，驱动模块需要满足高效稳定、低噪声等要求。因此，在设计电路时，需要根据不同的模块要求进行精心设计和布局，确保智能车的电路稳定可靠。

(3)软件设计部分主要基于速度和方向的控制设计，通过控制车模后轮驱动电机实现，同时辅以编码器传感器和舵机的转向控制。在设计软件时，需要考虑到控制算法的复杂性和实时性，需要使用高效的编程语言和算法来实现控制功能。同时，还需要考虑软件的可移植性和可维护性，确保智能车的软件稳定可靠。

机械设计、电路设计和软件设计等方面都是非常重要的一环，需要小心谨慎地完成。只有在各个方面都做得非常好，才能制作出高性能、稳定可靠的智能车。

第二章 结构设计

2.1 总体系统概述

基于 Infineon 的 TC264 单片机作为系统控制处理器，旨在设计一款智能自动驾驶小车。整个系统由主控模块、电机驱动模块、舵机转向模块和传感器模块组成。在传感器模块中，采用摄像头传感器获取赛道图像信息，该传感器可精准识别赛道中的各种元素，如直线、弯道、斑马线等[3]。同时，采用编码器传感器对电机速度进行检测，以及加速度陀螺仪传感器即姿态传感器检测是否遇到坡道，从而实现自动驾驶模式下车辆的稳定性和安全性。

在舵机转向模块中，采用模糊 PD 方式对舵机转向进行反馈控制。模糊控制是一种非精确控制方法，其基本思想是将输入量（这里是摄像头传感器获取的图像信息）映射到一组模糊语言变量上，然后通过一系列模糊规则进行模糊推理，最终输出舵机的控制信号。与传统的 PID 控制相比，模糊控制更加适合处理非线性、模糊的系统。

在电机驱动模块中，采用增量式 PID 算法完成对车速的闭环控制，并通过 PWM 控制驱动电路调整电机功率。PID 控制是目前控制工程中最常用的一种控制方式，它通过不断调整控制器的输出，使系统的实际输出尽可能地接近期望输出。本设计采用增量式 PID 算法，将控制量的变化量作为 PID 控制器的输入，从而实现车速的闭环控制，提高了小车的稳定性和精度[4]。

最后，通过软件设计的相关算法识别赛道各元素，并进行数据处理和控制指令的生成，从而实现小车的自动驾驶。系统整体结构框图如图 2-1 所示，各模块之间相互独立，但又相互关联，共同构成了一个完整的智能自动驾驶小车系统。

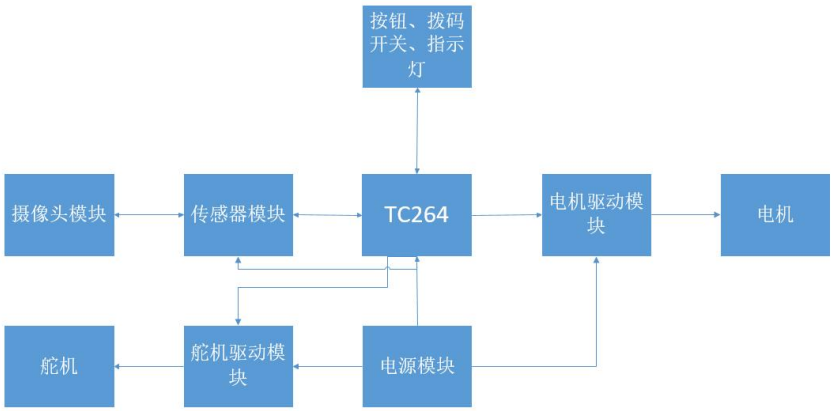

图 2-1　整体结构框图

2.2 车模结构布局

(1)将舵机直立安装，以提高其响应速度；

(2)将电池贴板放置在车身中后部，以降低小车重心；

(3)将摄像头前置，以便采集更多的赛道信息；

(4)降低整个车底盘的高度，以提高小车的行驶平稳性；

(5)将主板前置，驱动板后置，以便对电线进行整体规划。

制作完成后小车整体布局如图 2-2 所示：

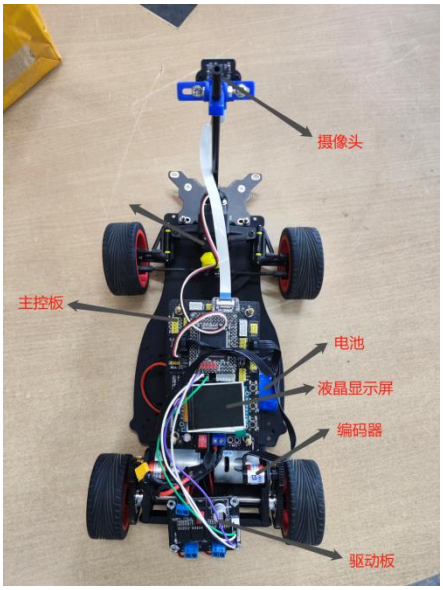

图 2-2　车模整体布局

2.3 车模类别介绍

此次竞赛指定的是 C 型车模，它采用双电机驱动结构，舵机控制转向，后轮电机通过差速调节实现转向。C 型车模的是能够通过软件方便地调节差速，但这也要求我们必须精确地感知小车在不同行驶状态下的差速需求。

C 型车模是全国大学生智能汽车竞赛中常用的一种车型，其参数和优势如下：

参数：车长：350mm、车宽：200mm、车高：130mm、轴距：230mm、轮距：165mm、重量：约 2kg。

优势：

(1)车身结构简单，易于制造和维护；

(2)可以配备了多个传感器，如红外线传感器、超声波传感器和陀螺仪等，能够实现多种功能，如避障、自动寻路等；

(3)车身比较轻便，动力系统采用直流无刷电机驱动，能够实现高速运动和快速控制；

(4)车身尺寸适中，能够在不同场地和环境中自如行驶。

缺点：

(1)因为车身结构简单，其稳定性和可靠性相对较差；

(2)由于传感器的种类和数量有限，其功能和性能相对于其他车型有所欠缺。

2.4 机械结构设计方案

C 型车模是全国大学生智能汽车竞赛中常用的一种车型。在比赛中，小车的运行状态是至关重要的。我们通过不断的摸索和调试发现，C 型车模的原有机械结构存在很多问题，这导致小车的运行状态非常差。因此，在规则允许的范围内，我们对整车的机械结构进行了系统的分析和反复的探索，进行了多方面的改进和调整，以提高小车的平稳性和整体精度。

(1)C 型车模的原始车身结构比较简单，重心较高，容易出现侧翻等问题。为了改善这一问题，采用了对称的车身设计，使得小车的重心更加稳定。此外，我们还对车身的材料和工艺进行了改进，以提高小车的结构强度和稳定性。

(2)对前轮的定位进行了优化。在原有结构中，前轮定位不够准确，导致小车在转弯时容易偏离轨道。为了解决这一问题，采用了更加精准的前轮定位方案，并进行了反复的调试和优化，以确保小车在转弯时能够保持稳定的运行状态。

(3)对舵机的灵敏度进行了调整。在原有结构中，舵机的灵敏度不够高，导致小车在转弯时反应缓慢，难以准确控制。为了修改这一问题，采用了更加灵敏的舵机，并进行了合理的调整，以确保小车能够快速、准确地响应指令，并保持稳定的运行状态。

通过以上的改进和调整，提高了 C 型车模的平稳性和整体精度。这次经历让我深刻认识到，对于智能汽车而言，机械结构是非常关键的一环，只有通过不断的优化和改进，才能使小车在复杂的环境中稳定、准确地运行。

1.前轮定位调整：

为了增加前轮的抓地力和转向对称性，在小车调试的过程中，可以通过调节主销内倾角、主销后倾角、前轮外倾角和前轮前束等来进行调整。然而，在小车调试的过程中，发现小车本身存在不对称问题，导致小车在正反向行驶时的转向不对称。为了解决这个问题，通过选择调整主销内倾角和前轮前束。需要注意的是，由于车模本身精度的限制，这部分角度的调整并不是主要因素，仅仅是为了避免负面影响并修正车模本身的不对称和不平衡问题。如图 2-3 所示。

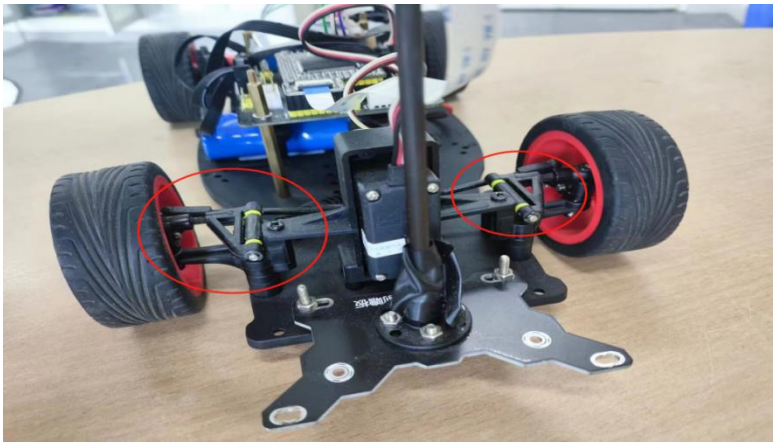

图 2-3 车模前轮定位

(1)主销内倾角：

主销内倾角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，但转向时越费力，轮胎磨损增大[5]。对于模型车，因此汽车的主销内倾角都有一个范围，约 5° —8° 之间。对于模型车，通过调整前桥的螺杆的长度可以改变主销内倾角的大小，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)前轮前束：

前轮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，作用是保证汽车的行驶性能，减少轮胎的磨损。内八字前端小后端大的称为“前束”， 反之则称为“后束”或“负前束”。前轮前束的前轮在滚动时，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[6]。在模型车中，前轮前束可通过改变转向横拉杆的长度实现。

2.5 舵机设计

舵机转向机构是小车行驶中极其重要的部分，舵机的安装位置和左右横拉杆的长度对舵机的灵敏度有着极大的影响。为了使舵机转向更加顺滑，选择将舵机架高，并在其两侧选用铜柱将其架高和加固。在舵机的上下部位加装了车模机械组装物件中的铁座，并用螺丝固定，以此起到舵机上下固定作用。需要注意的是，在安装铜柱时一定要保证左右的对称性，以保证舵机转向的精度。另一方面，为了提高舵机的反应灵敏度并使其在相同转角下有尽可能大的线行程，选择缩短了舵机的左右横拉杆。然而，舵机的左右横拉杆也不能太短，否则在相同的转矩下会影响其反应灵敏度。因此，经过实验后选择将其缩短至相对平行的状态下。这样做最大程度上减少了舵机转向时因左右横拉杆太长而损失的时间。如图 2-4 所示。

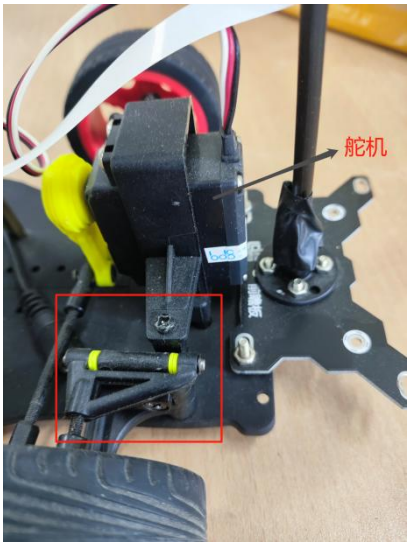

图 2-4　舵机的安装

2.6 传感器的设计和选择

传感器在车模中的作用非常重要，类型分别有视觉传感器、声学传感器、惯性传感器、磁性传感器，这些都可以帮助识别赛道信息和检测车速等。因此，在机械安装小车时，传感器的位置安装和固定等需要有整体的把控，经过考虑，选择出摄像头来识别感知周围赛道的地形，从而进行路径规划[7]。

2.6.1 摄像头的选择和设计

在选择摄像头时，主要从分辨率、动态特性、FPS 和自动曝光等参数来考虑。摄像头有数字和模拟两种类型，最终选择了总钻风 MT9V034 数字摄像头，因为它能够将图像的灰度值以数组的形式传输给单片机，并且与其他摄像头相比，具备更好的动态性能。

为了确保摄像头能够稳定循迹而不晃动，摄像头和底座的固定是至关重要的。因为竞赛要求摄像头镜面中心离地面不超过 15cm，所以需要先确定安装位置，然后再进行摄像头前瞻和视野宽度等方面的调整，以便采集赛道信息。为了固定摄像头，采用碳纤维杆制作摄像头支架，并通过打孔将铝制底座用螺丝固定在小车上。最后，使用胶水将支架和底座进行二次加固。如图 2-5 所示。

图 2-5　摄像头的安装

2.6.2 编码器的选择和设计

智能小车的运动快慢是通过速度来反映的。编码器作为速度检测反馈传感器，其安装位置需要考虑到测速的准确性。我们选择了龙邱科技的 1024 线编码器，因其体积小、稳定性高。最终，我们将其安装在小车的最末尾部分，以获得最准确的测速结果。如图 2-6 所示

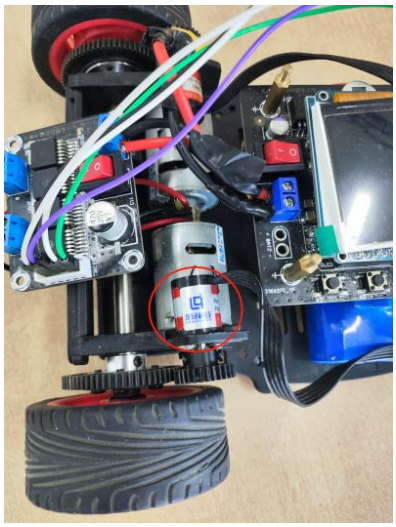

图 2-6　编码器的安装

2.7 电路板的设计及固定

小车上的电路板分别为主板和驱动板。根据电路板的设计和小车的整体布局考虑，选择的是将主板置于小车中前方和将驱动板后置。主板和驱动板选用胶柱架高并打孔后用螺丝固定。如图 2-7 所示。

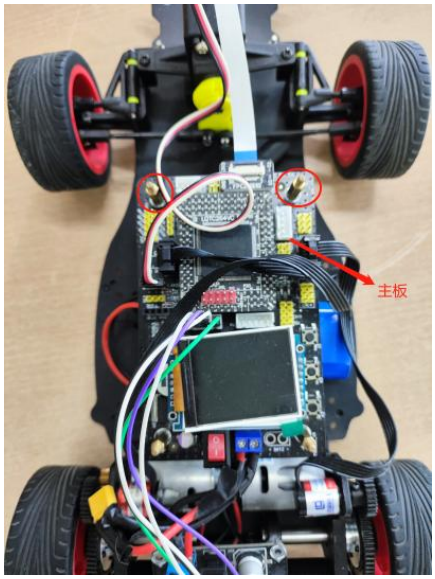

图 2-7 电路板的安装

第三章 硬件系统

3.1 硬件设计基础

智能汽车作为一种新型的智能工具，其核心部分是电路系统。在智能车的电路设计中，单片机最小系统、主控模块、传感器模块和驱动模块是不可或缺的组成部分。在硬件设计中，需要考虑到系统的稳定性、可靠性、高效性、简洁性和美观性等因素，以实现硬件电路的优化，从而保证智能车在各种复杂环境下的稳定运行和高效工作。

(1)稳定性和可靠性是电路设计的首要原则。在设计过程中，需要选用高品质的元器件，并采用合适的电路结构和布局，以确保电路的稳定性和可靠性。同时，还需要对电路进行充分的测试和调试，以发现和排除潜在的故障和问题。

(2)高效性也是电路设计的重要因素。在智能车的设计中，电路的响应速度和运行效率直接影响车辆的控制和运行。因此，在电路的设计中，需要优化电路的结构和算法，并采用高速的处理器和传感器，以实现电路的高效运行和快速响应。

(3)简洁性和美观性也是电路设计中需要考虑的因素。随着电子技术的不断发展，元器件的种类和数量不断增加，电路的复杂度也在逐渐提高。为了使电路的设计更加简洁、紧凑和美观，需要采用高密度的电路板，合理布局电路元器件，减少电路板的面积和元器件的数量，从而实现电路的简洁、美观和高效。

在智能车的电路设计中，还需要考虑到电路的可维护性和可升级性。电路的维护和升级需要耗费大量的时间和精力，因此需要在电路设计中充分考虑这些因素，并采用可拆卸、可替换的元器件和模块，以便于电路的维护和升级。

3.2 主控系统

3.2.1 电源模块

电源的可靠性决定着整个硬件系统的工作。电源供电不稳定可能会导致电池损耗、单片机复位、舵机和传感器损坏等严重问题[8]。因此，电源的设计是硬件电路设计中最为重要的部分之一。电源供电分配如图 3-1 所示。

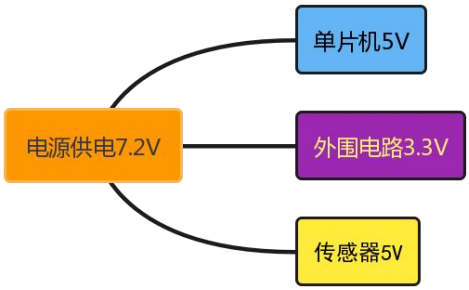

图 3-1　电源供电分配图

在经过考虑后，选择了 DC-DC 电源方案。芯片分别为 LM2596CS、LM39100。

LM2596CS 芯片，是一种高效率降压调节器芯片。LM2596CS 芯片的输入电压范围为 4V 到 40V，输出电压范围为 1.23V 到 37V 之间，输出电流可达 3A。该芯片采用了降压开关电源的技术，通过高效率的功率转换技术将输入电压降低到所需的输出电压，从而实现降压转换的功能[9]。此外，该芯片还具有过温保护、短路保护、过载保护等多种保护功能，可以有效保护电路免受损坏。

LM39100 芯片是一种高精度电流调节器芯片，它是一种常数电流输出的多功能电流源，可广泛应用于电子设备中。LM39100 芯片具有多种保护功能，包括短路保护、过温保护、过压保护等，可以保护电路免受损坏。此外，该芯片还具有高精度的电流输出、宽输入电压范围、低温漂移和低噪声等特点。LM39100 芯片的输入电压范围为 2.7V 到 7V，输出电流可达 1A，适用于需要高精度电流输出的各种应用场合，如电源模块、LED 照明等。

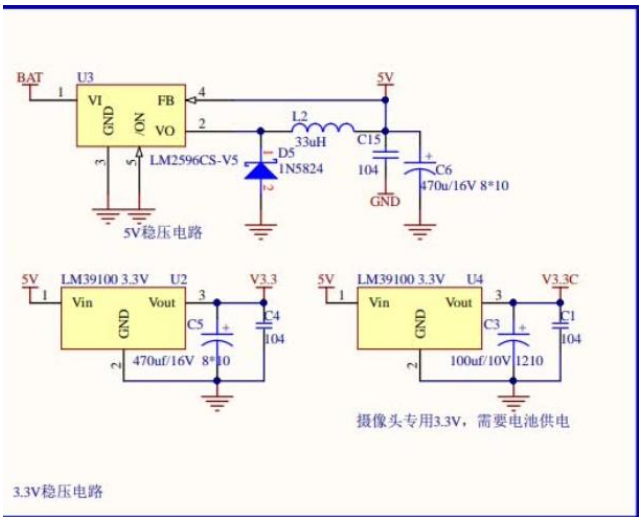

图 3-2　5V 和 3.3V 稳压电路

通过使用 LM2596CS 电源芯片将设备供电 5V，这款芯片具有完善的保护电路和较好的稳压性能，即使电池电压较低也能保证单片机和传感器正常工作。为了减少纹波对元件的影响，我们在电源输出位置使用电容进行滤波。同时，电机驱动部分采用直接接入电源的方式。而对于 3.3V 电源供电，选择了 LM39100 稳压芯片。5V 和 3.3V 电源供电电路原理图如图 3-2 所示。

3.2.2 蜂鸣器模块

蜂鸣器电路原理图如图 3-3 所示

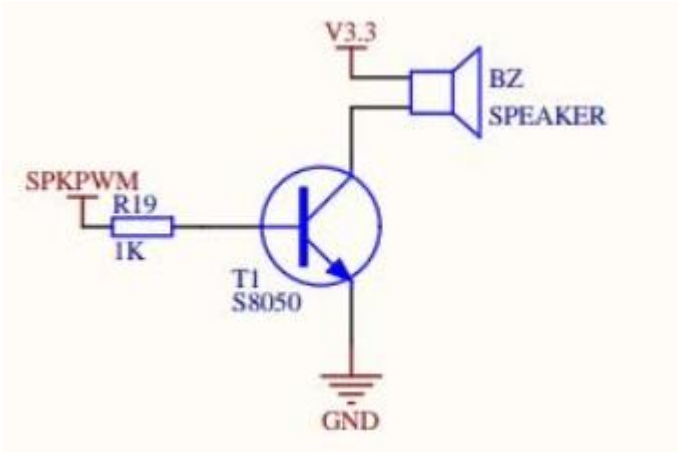

图 3-3　蜂鸣器电路

3.3 传感器检测系统

智能车用到的传感器主要有摄像头传感器、编码器传感器。

3.3.1 摄像头传感器

摄像头选用的是龙邱科技的 MT9V034 总钻风摄像头，其用来检测和采集赛道信息。所以在进行电路设计时，需要结合赛道情况尽可能通过硬件保证检测的有效性和可靠性。摄像头传感器电路原理图如图 3-4 所示。

图中各个引脚的作用：GND：地线；D2-D9：存储摄像头数据；SDA：I2C 总线数据引脚，用于芯片的配置与控制；SCL：I2C 总线时钟引脚，用于芯片的配置与控制；PLXCLK: 像素时钟引脚，用于同步数据输出；HREF：行同步引脚，用于同步每一行像素的输出；RESET：复位引脚，低电平有效；VSY：场同步引脚，用于同步每一场像素的输出；3V3：接 3V3。

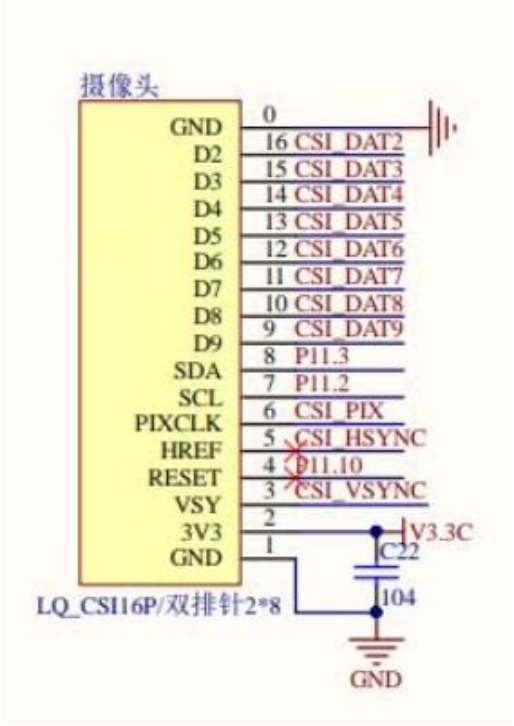

图 3-4　摄像头接口

3.3.2 编码器传感器

编码器选用的是龙邱科技的 1024 线编码器传感器，其用来进行速度检测反馈。其电路原理图如图 3-5 所示。

图中各个引脚的作用：1：地线；2：接 3v3；3：表示编码器的 A 相输出信号，可以通过检测它的高低电平变化来确定编码器的旋转方向和角度；4：表示编码器的 B 相输出信号，与 A 相信号一起使用可以提高编码器的分辨率。

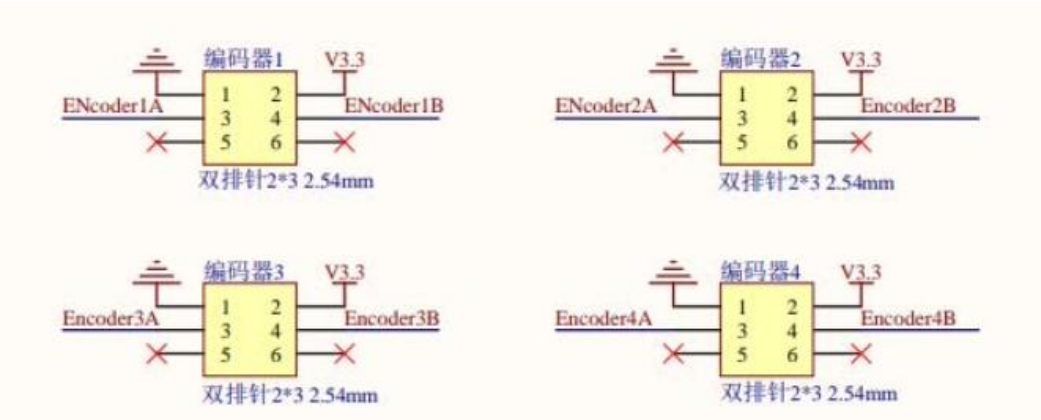

图 3-5 编码器接口

3.4 电机驱动模块

驱动电路为智能车驱动电机提供控制和驱动，因此电机驱动的设计是硬件电路中非常重要的一部分。驱动电路的基本原理时 H 桥驱动原理。驱动电路主要由隔离电路、栅极驱动电路、保护电路和 H 桥组成[10]。驱动模块其中隔离原理图如图 3-6 所示。

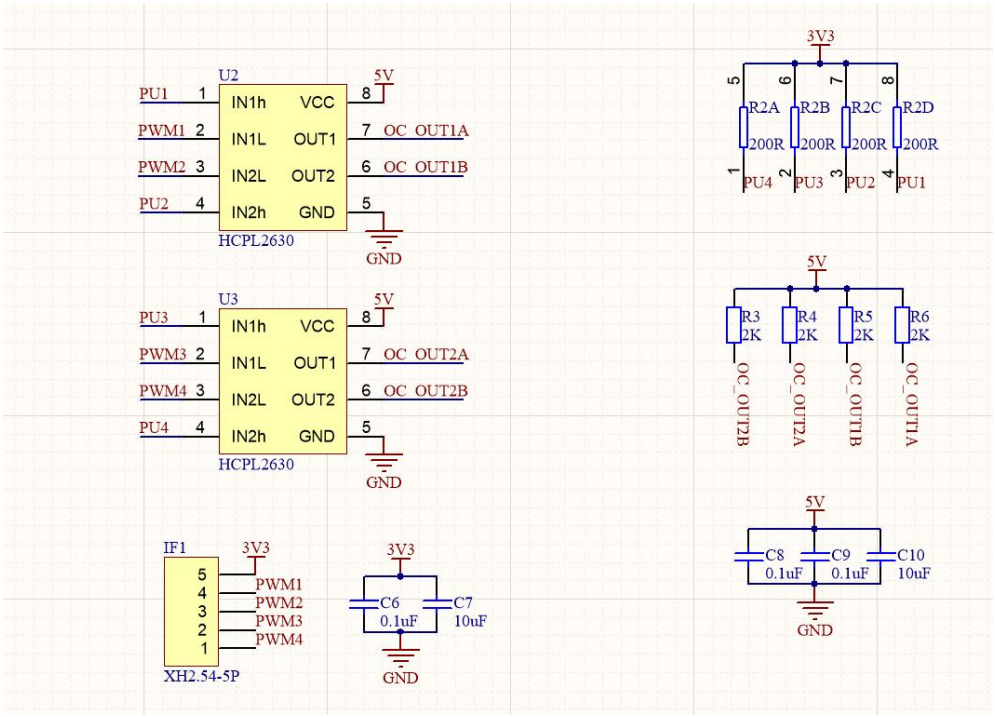

图 3-6　驱动电路隔离原理图

第四章 软件系统设计

4.1 软件设计总体框架

智能车的核心是其控制系统，也就是车辆的程序。智能车的程序可以通过一个完整稳定的系统软件控制框图实现易读性和高可靠性。系统控制部分包括速度控制和方向控制，这两个控制系统是智能车能够安全、高效运行的基础。其中，速度控制主要通过控制车模的后轮驱动电机实现。后轮驱动电机是智能车的动力源，通过对电机的控制，智能车可以根据需要调整自己的速度。而方向控制则需要舵机和后轮差速器的相互协同配合。舵机可以控制车模的转向，而后轮差速器则可以调整车轮的差速，以实现更加精准的转向控制。通过软件相关算法，系统可以最终完成对小车的速度和方向控制。在智能车的程序开发中，易读性和高可靠性是非常重要的。易读性是指程序的代码应该能够被开发人员轻松理解和修改，以便快速解决问题。而高可靠性则是指程序应该尽可能少出现错误，保证智能车的稳定性和安全性。为了实现这两个目标，智能车的程序需要遵循一些最佳实践，例如采用模块化编程，使用注释和文档记录代码等。此外，在程序开发过程中，还需要进行充分的测试和验证。测试可以发现程序中存在的问题和潜在的错误，验证可以证明程序的正确性和可靠性。在测试和验证过程中，可以采用各种测试工具和模拟环境来模拟实际运行情况，以确保程序的稳定性和可靠性。软件控制设计框图如图 4-1 所示。

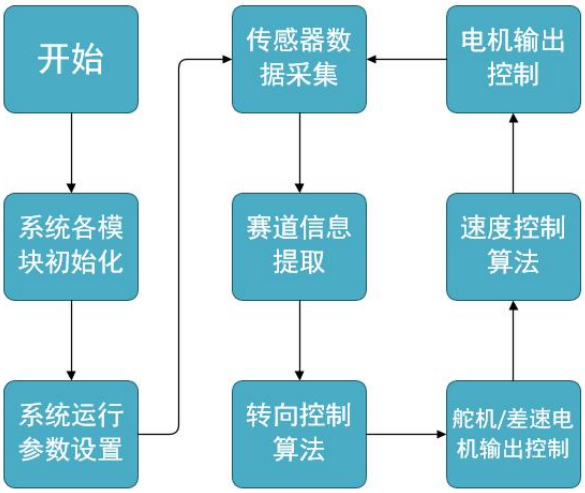

图 4-1　软件控制设计框图

4.2 系统各模块初始化

为了让系统中的每个模块（例如摄像头和电机）在使用前都处于正常状态，需要对它们进行初始化。为了实现这个目标，将所有的初始化程序汇总在一起，并采用的四核 TC264 中的 0 核和 2 核分别处理图像和其他任务。因此，需要创建了两个初始化子函数 Systeminit0()和 Systeminit2()，并在这些子函数中调用各个模块的初始化函数。

4.3 路径识别

4.3.1 图像处理

通过使用作为赛道信息识别传感器，通过返回彩色或灰度图像数据的像素比例来进行道路识别。该数字灰度摄像头在图像采集方面具有良好的稳定性，采用二维数组存储赛道信息，灰度值在 0-255 之间，颜色越偏白，灰度值越大，反之越小。摄像头的图像采集思路如下：

(1)定义行和场两个中断函数。

(2)当场中断发生时，开启行中断。

(3)在行中断中循环采集每行的像素点，可以设置跨行采集。

(4)当实际采集的行数等于定义的总行数时，表示图像采集完成，关闭行中断，将采集的数据发送至上位机。

4.3.2 循迹处理

基本图像处理完成后，需要从图像中提取出有效的赛道信息，即赛道边缘，以实现小车的行驶路径和特殊元素的识别。在直道和弯道上，利用摄像头输出的图像经过灰度化处理后，可以得到黑白二值化的图像。通过对二值化图像进行处理，可以检测出黑色线条的位置。智能车上的控制系统可以根据检测到的线条的位置，控制车轮的转向，使车辆沿着黑色线条行驶。而对于特殊元素的循迹思路如下：

(1)对于十字回环,由于只能搜到赛道一边的拐点，确定找点的初始位置，选择另一边进行补线处理，然后给左右面积乘以某个系数，同时根据矫正图像滤除十字回环拐角线，从而顺利连线；

(2)对于圆环，能够找到圆环的一边边界的列数会有较大的差异，从而补线让其进入圆环内；

(3)对于坡道，分成两端来判断，远端的右边线与左边线的差之和大于近端的右边线与左边线的差之和，便会识别坡道，坡道会采用减速，并将中线算出来的舵机转角除以一个值；

(4)对于三岔路口，首先判断左右直线三种进入状态，并通过白点长度找到中间的最近点，然后将它和需要转向的方向的反方向的边线连接起来，使其转向[11]；

(5)对于车库，出库时，用来判断赛道的长度来控制直行与转向；入库时，检测到起跑线后直行一定距离后转向入库。

4.4 速度控制

通过分析提取的赛道信息，可以计算出理想速度，并使用 PID 算法根据理想速度计算输出到电机的 PWM 波的占空比大小，以控制速度。例如，在直道上，智能车的速度应该快速提升至用户设定的最大速度，而在弯道上，智能车应该及时将速度调整到用户设定的最小速度。

方向控制的理论分析：

PID 控制是工业过程控制中历史最悠久，生命力最强的控制方式。这主要是因为这种控制方式具有直观、实现简单和调整方便等一系列的优点。PID 控制主要有三部分组成，比例单元（P）、积分单元（I）、微分单元（D）。比例控制是一种最简单的控制方式。其控制器的输出与输入误差信号成比例关系。偏差一旦产生，调节器立即产生控制作用使被控量朝着减小偏差的方向变化，控制作用的强弱取决于 KP。 当仅有比例控制时系统输出存在稳态误差为了消除稳态误差，引入积分控制。积分项对误差取决于时间的积分，随着时间的增加，积分项会增大。这样，即便误差很小，积分项也会随着时间的增加而加大，它推动控制器的输出增大使稳态误差进一步减小，直到等于零。为了预测预测误差变化的趋势，引入微分的控制器，这样就能 够提前使抑制误差的控制作用等于零，甚至为负值，从而避免了被控量的严重超调[12]。增量式 PID 算法是一种基于比例-积分-微分控制器的控制算法，其主要作用是根据系统的误差和误差变化率来计算出控制量的增量，从而实现对系统的控制。相对于标准的 PID 控制器，增量式 PID 算法不需要存储历史数据，使得其计算效率更高，同时也能更好的应对系统参数变化和干扰。

PID 控制器是一种线性控制器，它根据给定值与实际输出值构成控制偏差，将偏差的比例(P)、积分(I)和微分(D)通过线性组合构成控制量，对被控对象进行控制。原理框图如图 4-2 所示。

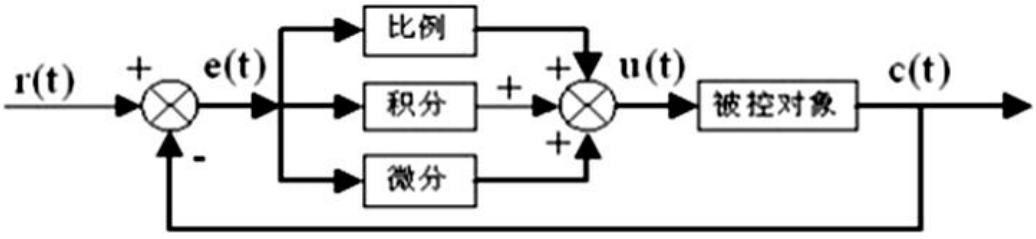

图 4-2 PID 控制器原理

4.5ADS 软件安装、调试

4.5.1 软件编译环境

开发工具使用的是 AURIX Development Studio，简称 ADS。这个 IDE 包含调试环境，不需要许可证，而且配置好了启动的工程，使用起来更加方便。在目标程序的下载方面，通过下载器与单片机之间的连接下载程序。其软件界面如图 4-3 所示。

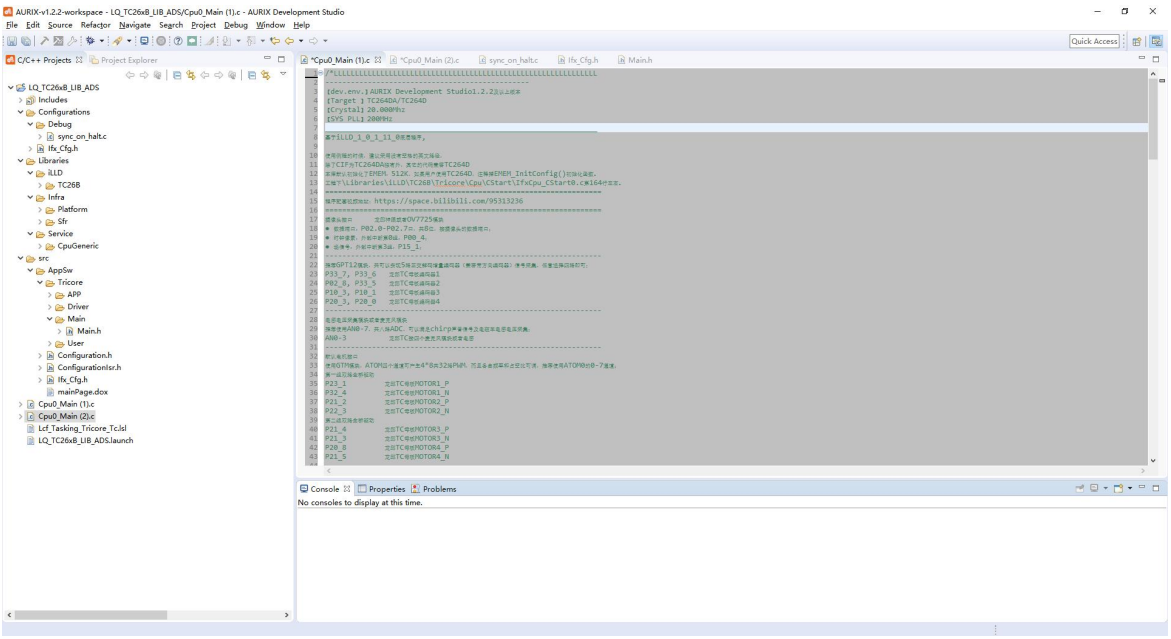

图 4-3　ADS 开发环境界面

4.5.2 调试

我们采用了 OLED 作为显示模块，用于辅助调试。相比传统 LCD 显示屏，OLED 无需背光源，更轻更薄，可视角度更大，柔软环保且更省电。在调试前期，不得不重新下载程序以更改数据，这显得非常繁琐且低效。为了解决这个问题，花费了很多功夫编写了一个液晶程序。现在，可以通过 OLED 的多级菜单实现对电感原始数据、标志位和数值的监测，对小车的 PID 参数进行按键调节并将数据写入 flash 中，从而方便了调试工作。

4.5.3ADS 编译环境

编译环境界面介绍 在打开工程模板之后就是 ADS 编译环境的界面，整体界面分为四部分， 最上面是 ADS 的工具栏、菜单栏，中间左侧是工作区，中间右侧是代码编辑区，下面是编译输出。具体功能如图 4-4 所示

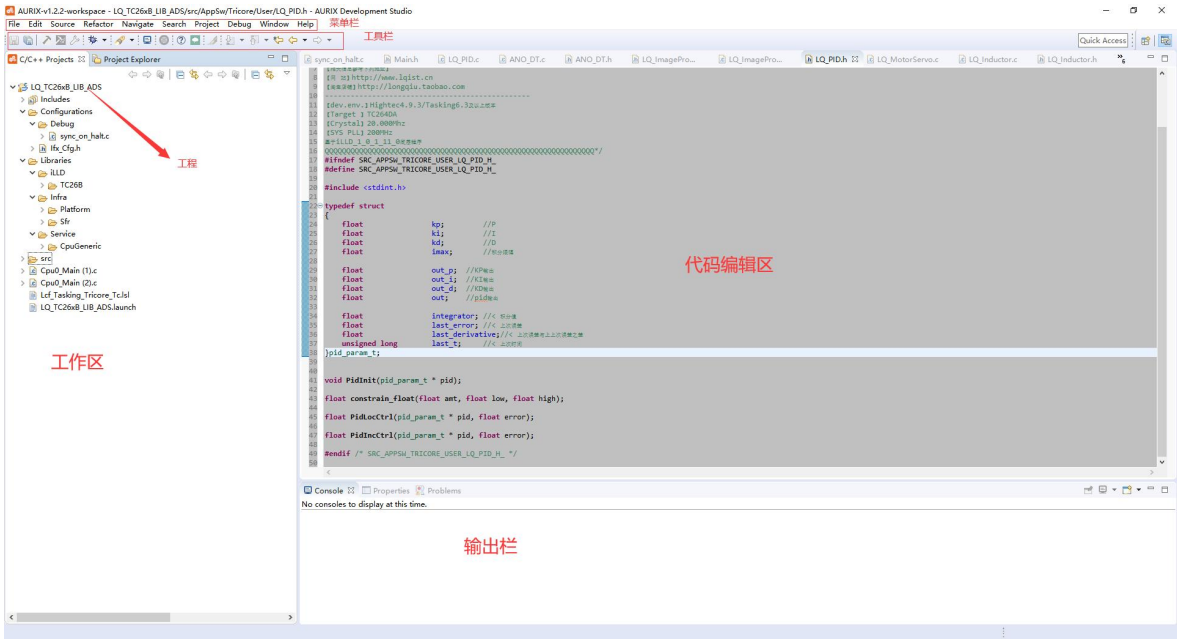

图 4-4 各个区域的功能

建立 ADS 工程过程：

(1)选择 File（文件）菜单，然后选择 New（新建）。

(2)在弹出的 New Project（新建工程）对话框中，选择要创建的工程类型。例如，可以选择 Automotive AURIX C/C++ Project（汽车 AURIX C/C++工程）。

(3)在下一个对话框中，输入工程名称和路径，选择工具链（Toolchain）、处理器类型（Device Variant）和 Flash 算法（Flash Algorithm）等选项。然后点击 Next（下一步）按钮。

(4)在下一个对话框中，选择要添加到工程中的源文件和头文件。可以选择已有的文件，也可以创建新的文件。然后点击 Finish（完成）按钮。

5.这样就建立了新工程。可以在 Project Explorer 窗口中看到新建的工程及其组成部分。现在可以开始编写代码和调试程序了。

第五章 智能车调试结果

5.1 智能汽车技术参数

传感器种类及数量：摄像头*1、编码器*2

电池容量：2000 mAh

5.2 智能汽车外形参数

经过改装后，智能汽车的外形参数为：

车长：432mm；

车宽：260mm；

车高：160mm；

车重：1265g

5.3 智能汽车完成结果及分析

使智能车在赛道进行实际测试，选取三次最好完成成绩，智能车完成全程平均时间为 29.4s，平均速度为 1.26m/s，赛道全长为 37.1m。测试结果如表 5-3 所示。

表 5-3 测试结果

	完成全程时间	平均速度
第一次	28.6s	1.29m/s
第二次	29.4s	1.26m/s
第三次	30.2s	1.23m/s

5.4 智能车通过特殊道路测试

智能车在赛道上行驶过程中可能会出现各种各样的情况，为解决这些情况，进行智能车在各个特殊元素（十字回环、圆环、坡道、三岔路口）的道路测试。

1.智能车通过在十字回环和圆环时，(十字回环如图 5-1 所示,圆环如图 5-2 所示)可能会出现两种情况：第一种是速度过快而冲出赛道，第二种是陷入循环转圈的情况。为避免情况的发生，解决方案如下：

(1) 控制智能车的速度是非常重要的。当智能车在十字回环中行驶时，速度过快容易造成惯性力过大，导致智能车无法准确判断路况，从而失控冲出赛道。所以，需要适当降低智能车的速度，以提高行驶安全性。

(2) 设置好判断点进行补线是保证智能车顺利通过十字回环的关键。在十字回环中，

智能车需要通过摄像头对路面进行实时监测，并根据监测结果进行补线。通过设置好判断点，可以准确地确定智能车的位置和方向，从而对路况进行及时的补线操作，确保智能车顺利通过十字回环。

(3) 调整摄像头位置高度和程序中参数也是保证智能车正常出入十字回环的重要因素。通过调整摄像头位置高度可以提高摄像头的拍摄角度和视野，从而更好地监测路面情况。同时，调整程序中参数也可以提高智能车控制的精度和稳定性，保证智能车在十字回环中行驶的正常性。

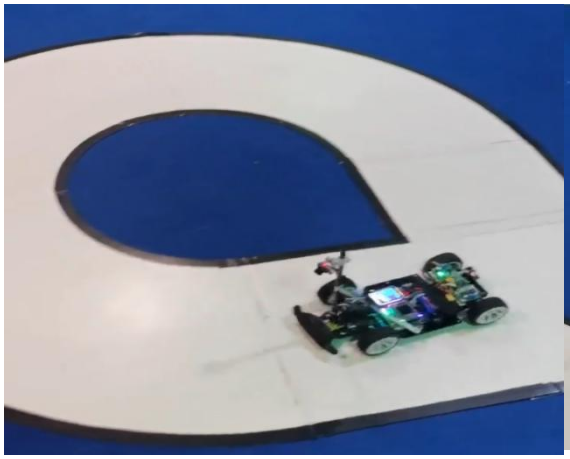

图 5-1 十字回环

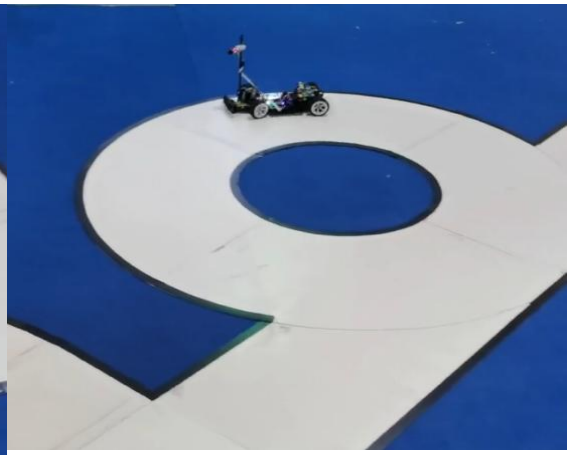

图 5-2 圆环

2. 智能车在通过坡道时，(坡道如图 5-3 所示)可能会有两种情况：第一种是智能车出现上不去坡道；第二种是智能车到半坡卡住的情况。为避免情况的发生，解决方案如下：

增加智能车的上坡速度,使智能车在行驶中根据路面情况来调整上坡速度。若速度设置过低，智能车将无法成功通过坡道；若速度过高，则可能会出现失控的情况。需要适当增加智能车的上坡速度，以提高行驶效率和上坡成功率。

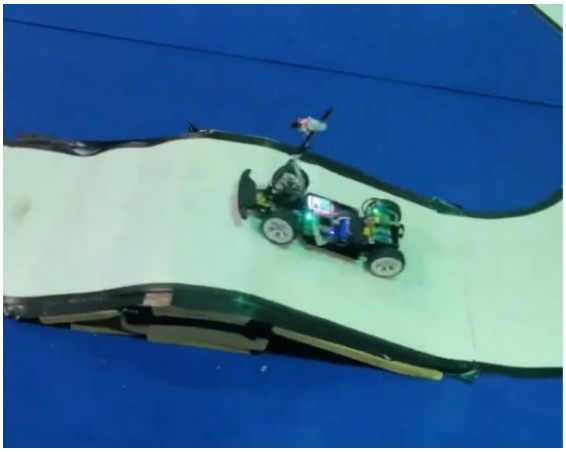

图 5-3 坡道

3. 智能车在通过三岔路口时，(三岔路口如图 5-4 所示)可能会出现三种情况：第一种是出现直接冲出路口；第二种是转向方向错误；第三种是出不去往回走的现象。为避免情况的发生，解决方案如下：

(1)调整摄像头位置高度。通过调整摄像头位置高度可以提高摄像头的拍摄角度和视野，从而更好地监测路口情况。这样可以准确地识别路口的结构和方向，从而避免出现转向方向错误的现象。

(2)修改程序中路口判断点问题。智能车在行驶过程中需要通过摄像头对路面进行实时监测，并根据监测结果进行补线和判断路口方向。如果路口判断点设置不当，就容易出现直接冲出路口或者出不去往回走的现象。所以，需要对程序中路口判断点进行调整和优化，以确保智能车能够准确地识别路口方向和行驶路径。

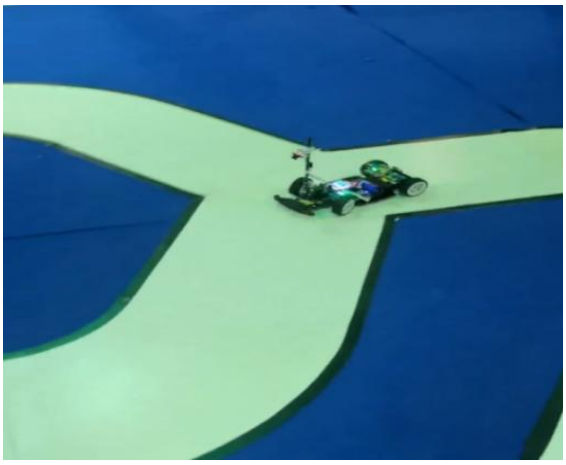

图 5-4 三岔路口

第六章 总结与展望

6.1 总结与不足

智能车的设计与调试需要学习机械、电子、计算机等多个领域的知识，这让我对智能车的设计和调试有了更深入的了解。在这个过程中，会遇到了各种各样的问题，例如机械结构的设计问题、电路板的焊接问题，程序调试问题等等。通过查询文献，与导师、同学的讨论，来解决这些问题。有了这次经历，我学会了如何解决问题，如何寻找问题的根源，以及如何采取有效的措施解决问题，从而使智能车能够顺利完成目标要求。智能车设计是一个漫长而艰苦的过程，需要付出大量的时间和精力，在控制算法上，算法完善可能会有所不足，但是，只要坚持不懈，就一定能达成目标需求。同时，我也要继续努力学习，尝试不同的算法，并与硬件方面相适配，进而使智能车更好的完成需求。

6.2 展望

(1)未来可以将智能汽车技术应用于城市交通体系，实现更加智能化、高效化的交通管理。

(2)可以研究智能车的自主决策能力和应对突发状况的能力，提高智能车的实际应用价值。

(3)可以改进智能车的传感器系统，提高循迹和速度控制的精度和稳定性。

(4)可以将智能车应用于更广泛的领域，如智能物流、无人驾驶等，推动智能化技术的发展和应用。

参考文献

[1] 张晓冬,王海燕,李继光,沈岩.汽车模块化思想在“飞思卡尔”智能车竞赛中的应用[J].中国现代教育装备,2015(05)

[2] 郭勤,蔡亚永.智能小车控制系统的设计与实现[J].电子测试,2021(11)

[3] 蒋贤海,罗彤.路径识别智能小车控制系统的设计与开发[J]. 广东水利电力职业技术学院学报. 2011(01)

[4] 邓良益,易佳,王浩,颜雁军,单飞桥.一种基于 PID 算法的智能小车设计[J].科技创新与应用,2019(30)

[5] 张遥远,刘美生,罗发贵,王蛟茹.四轮定位仪的性能评价与检测技术研究[J].中国测试技术,2004(04):3-6.

[6] 唐宏乾,张洲.简析汽车四轮定位[J].湖南农机,2012,39(03):100-101.

[7] Bao Fa Sun, . Software Design of Smart Car Tracking with Camera[J].Applied Mechanics and Materials, 2013, 2617(380384):2619-2622.

[8] 杨丰硕,钱靖宇,王浩.基于 STC8A 微控制器的节能车模系统[J].电子产品世界,2021,28(01):92-94.

[9] 张德盛. 基于 ARM 的数字式比例放大器的研究[D].浙江大学,2012.

[10] 郑子栋,梅孝安,蒋海峰,刘根,周双喜,田芃.基于线性 CCD 图像识别智能小车的设计与开发[J].电子技术,2015,44(02):62-64.

[11] San mao He, Dan feng Chen, Yu ping Cai, et al. Design and Implementation of the Maze Patrolling with Three Branch Roads by the Smart Car Based on 51 Single Chip Microcomputer[J]. Automation, Control and Intelligent Systems, 2020, 8(1)

[12] 赵龙彪,高亮,陈志敏,邱浩波.基于比例微分优化准则的拓扑优化方法[J].中国机械工程,2011,22(03):345-350.

附　录一 中文译文及外文资料

智能汽车摄像头跟踪软件设计

Software Design of Smart Car Tracking with Camera

摘要：

本文描述了带摄像头的智能汽车跟踪的软件设计思路。讨论了以下模块：主要功能结构、缓存模块的初始化、道路信息收集模块、道路信息处理模块、舵机控制模块和电机控制模块。智能汽车的反复调试和实际操作表明，它能够准确地收集道路信息，灵活控制，运行稳定，速度达到 2.2m/s。

正文：

本文将详细描述智能汽车跟踪摄像机的硬件设计方法，并探讨智能汽车的软件系统设计。智能汽车的硬件系统由电源管理模块、信息收集模块、数据处理模块、运动控制模块和信道通信模块组成。这些模块相互协作，以实现智能汽车的跟踪和移动。其中，信息收集模块用 CMOS 数字摄像机 OV6620 进行路面图像信息采集，数据处理模块对采集的图像数据进行处理，运动控制模块控制汽车的运动轨迹，电源管理模块则负责为各模块提供电源，信道通信模块则负责与其他设备进行通信。除了硬件系统外，智能汽车还需要相应的软件系统，即控制程序。因此，在设计硬件系统之后，我们需要设计智能汽车的软件系统。智能汽车软件系统的设计思路，主要集中在主要功能结构、每个模块的初始化、道路图像信息收集模块、道路信息处理模块、舵机控制模块和电机控制模块等方面。在主要功能结构方面，智能汽车的软件系统需要实现图像采集、处理、控制和通信等主要功能。每个模块的初始化是控制程序的关键部分，它能够保证硬件系统的正常运行。道路图像信息收集模块需要对路面图像进行采集和处理，以提取出行驶道路的信息。道路信息处理模块则根据采集到的信息进行处理，以确定汽车的行驶方向和轨迹。舵机控制模块根据处理后的信息对汽车的舵机进行控制，以确保汽车沿着预定的轨迹行驶。电机控制模块则根据舵机控制模块的指令对电机进行控制，以使汽车沿着预定的轨迹行驶。在软件系统设计中，我们还需要考虑对算法的选择和优化。例如，我们可以使用 PID 算法实现电机的闭环控制，以提高汽车的精度和稳定性。此外，我们还可以使用 PD 算法对舵机进行控制，优化汽车的转弯效果。总之，智能汽车的跟踪摄像机是一项复杂的技术，它需要完善的硬件系统和软件系统来实现。

智能汽车软件系统架构：该汽车使用 16 位微处理器 MC9S12XS128 作为核心控制芯片，使用 CMOS 数字摄像机 OV6620 收集路面图像信息，应用 PD 算法控制转向齿轮，应用 PID 算法利用光电编码器实现电机的闭环控制。在 MCU 的协调下，汽车可以沿着指定的车道快速行驶。A.主程序的结构：智能汽车的程序流程图如图 1 所示。根据程序的步

骤，所有模块的函数声明都列在主函数中，包括每个模块的初始化、道路信息检测模块、道路信息处理模块、方向控制模块和电机控制模块。主函数的代码如下：

```
 void main(void) //  相位锁定环的初始化
PLL Init();   //  中断捕获的初始化
IC Init(0);   // XS1280 的初始化
XS1280 Init();   //  串行通信接口的初始化
SCI Init();   //  脉宽调制的初始化
PWM Init();   //  启用中断
EnableInterrupts;   //  延迟
 delay();
for(;;)
 {
 //  图像二值化
void lmage binaryzation(unsigned int row);   //  边界提取，道路确定
Boundary cxtraction;
bianjic();
 panduan(); }
```

ABSTRACT：

The software desien ideas of smart car tracking with camcra are described. The followinmodules are discussed: the main function structure the initialization of cach modulc. roainformation collecting module, road information processing module, servo control module andmotor control module. The repeated debugging and the real operation of the smart car show that ican accurately collect the road information, control flexibly, run stably, and its velocity reache2.2m/s.

TEXT:

The hardware design method of smart car tracking with camera was described in literature (1]The hardware system of the smart car consisted of five parts: power management modulinformation collecting modulc, dat processng modulc, motion control module and scnacommunication module. A smart car must have hardware system and software system a car only with the hardwarcsystem but without the corresponding software system is not a smart car. Therefore, we need todesien the sotware systcm of the smart car after desemine the hardware system. namcly. destan thcontrol prouram of he smart car 2.3.4This paper introduces the design ideas of the software systemm of the smart car, focusing on themamn function structure, the

initialization of cach module, road image information collecting modulroad information processing module, servo control module and motor control module, ctc| 5).

The Software System Arehitecture of the Smart CarThe car uses 16 bits microproccssor MC9S12XS128 as the kernel control chip. USs CMOSdigital camcra OV6620 to collect the road image information, applies PD algorithm to control thestecring gcar, apples PID alzorithm to impement the closed-loop control of the motor with the helrof photoelectrie encoder.Under the MCU's coordination of each part the car can run fast along the specified lancA.The Structure ofthe Main Program The program flow diagram of the smart car is shown in Fig. 1In accordance with the procedures, function declarations of all modules are listed in the mairfunction. mcluding the iitialization of cach modulc, road informmaton detecting modulc, roadinformation processing module, direction control module and motor control module.The code ofthe main function is as following:

```
yoid main(void)
// Initialization of the phasc-locked loop
  PLL Init();
IC Init(0):
// Initialization of interruption capture
XS128 0 Init();
// Initialization of inputoutput port
SCI Init();
//Initialization of serial communication intertacc
/ Initialization of Pulsc-Width Modulation
PWM Init():
EnableInterrupts:
/Enable interruption
delay();
for(::)
{ void lmage binaryzation(unsigned int row)/ Binaryzation of the image dataBoundary cxtraction// Road determination,
bianjic();
panduan();
```

附　录二 部分程序及初始化代码

```
void TFT_Show_Camera_Info (void)
{
    char txt[16] = "X:";

    sint16 mps = 0, dmm = 0;       // 速度：m/s,毫米数值
    sint16 pulse100 = 0;
    uint16 bat = 0;



    dmm = (sint16) (RAllPulse * 100 / 579);              // 龙邱 512 带方向编码器 1 米 5790 个脉冲，数值太大，除以 100
    pulse100 = (sint16) (RAllPulse / 100);
    sprintf(txt, "AP:%03d", my_piancha);                 //
    TFTSPI_P8X16Str(3, 4, txt, u16RED, u16BLUE);     // 显示赛道偏差参数

    NowTime = (STM_GetNowUs(STM0) - NowTime) / 1000;   // 获取 STM0 当前时间，得到毫秒
    mps = (sint16) (dmm / (NowTime / 1000));                 // 计算速度 mm/s
    // TFTSPI_Road(18, 0, LCDH, LCDW, (unsigned char *)Image_Use); // TFT1.8 动态显示摄像头灰度图像
    TFTSPI_BinRoad(18, 0, LCDH, LCDW, (unsigned char *) Bin_Image);    // TFT1.8 动态显示摄像头二进制图像
    sprintf(txt, "%04d,%04d,%04d", duandian, currentzhongjian[30], currentzhongjian[45]);
    TFTSPI_P8X16Str(0, 5, txt, u16RED, u16BLUE);           // 显示赛道偏差参数
    BatVolt          = ADC_Read(ADC7);   // 刷新电池电压
    bat = BatVolt * 11 / 25;    // x/4095*3.3*100*5.7
    sprintf(txt, "B:%d.%02dV %d.%02dm/s", bat / 100, bat % 100, mps / 1000, (mps / 10) % 100);    // *3.3/4095*3
    TFTSPI_P8X16Str(0, 6, txt, u16WHITE, u16BLUE);     // 字符串显示
    // 电机和舵机参数显示
    sprintf(txt, "Sv:%04d Rno:%d", ServoDuty, CircleNumber);
```

```
    TFTSPI_P8X16Str(1, 7, txt, u16RED, u16BLUE);        // 显示舵机，电机 1，编码器 1
数值
    sprintf(txt, "AD0:%04d, AD1:%04d ", Lleft1, Lleft2);
    TFTSPI_P8X16Str(0, 8, txt, u16RED, u16BLUE);        // 左侧电感数值
    sprintf(txt, "AD2:%04d, AD3:%04d ", Lright1, Lright2);
    TFTSPI_P8X16Str(0, 9, txt, u16RED, u16BLUE);        // 右侧电感数值
    /*
    sprintf(txt, "Ring num: %d ", CircleNumber);
    TFTSPI_P8X16Str(2, 6, txt, u16GREEN, u16BLACK);
    sprintf(txt, "M:%03d Q:%d J:%d ", MagneticField, TangentPoint, EnterCircle);
    TFTSPI_P8X16Str(0, 7, txt, u16WHITE, u16BLACK);
    */
}
/*************************************************************************
 *  函数名称：void CameraCar(void)
 *  功能说明：电磁车双电机差速控制
 *  参数说明：无
 *  函数返回：无
 *  备      注：驱动 2 个电机
 *************************************************************************/
void CameraCar (void)
{
    // 摄像头初始化
    CAMERA_Init(50);
//    RAllPulse = 0;                          // 全局变量，脉冲计数总数
    NowTime = STM_GetNowUs(STM0);    // 获取 STM0 当前时间
    while (1)
    {
        // LED_Ctrl(LED1, RVS);        // LED 闪烁 指示程序运行状态
         if (Camera_Flag == 2)
         {
              Camera_Flag = 0;        // 清除摄像头采集完成标志位   如果不清除，则不会
再次采集数据
```

```
                Get_Use_Image();        // 取出赛道及显示所需图像数据
                Get_Bin_Image(0);       // 转换为 01 格式数据，0、1 原图；2、3 边沿提取
                Bin_Image_Filter();  // 滤波，三面被围的数据将被修改为同一数值
            //    Seek_Road();              // 通过黑白区域面积差计算赛道偏差值

                // 计算赛道偏差值，系数越大打角越早，数值跟舵机的范围有关，此处为±160 左右，默认为 7，
            //     ServoDuty = Servo_Center_Mid - (OFFSET1 + OFFSET2 + OFFSET2) * 1 / 7;

                // 圆环处理，如果面积为负数，数值越大说明越偏左边；
//                  if((OFFSET2 < -300)||(OFFSET2 > 300))
//                      ServoDuty = Servo_Center_Mid - OFFSET2 / 7;

//                  ServoCtrl(ServoDuty);        // 舵机 PWM 输出，转向

                // SPEED 正负标识方向，负数为正向
//                    MotorDuty1 = MtTargetDuty + ECPULSE1 * 4 - (OFFSET1 + OFFSET2 + OFFSET2) / 10;            // 电机 PWM
//                    MotorDuty2 = MtTargetDuty - ECPULSE2 * 4 + (OFFSET1 + OFFSET2 + OFFSET2) / 10;            // 双电机差分，需要去掉 abs

                // MotorCtrl(MotorDuty1, MotorDuty2);            // 四轮电机驱动
                //   MotorCtrl(1000, 0); // 电机 PWM 固定功率输出
            }
//              if(RAllPulse>10000)
//                  Reed_Init();                // 干簧管 GPIO 及中断初始化函数,为停车入库做准备
//              if (Game_Over)
//              {
//                  OutInGarage(IN_GARAGE, ReadOutInGarageMode());
//              }
```

```
    }
}
#ifndef SRC_APPSW_TRICORE_USER_LQ_IMAGEPROCESS_H_
#define SRC_APPSW_TRICORE_USER_LQ_IMAGEPROCESS_H_


void CameraCar(void);
void TFT_Show_Camera_Info(void);

#endif /* SRC_APPSW_TRICORE_USER_LQ_IMAGEPROCESS_H_ */
sint16 TempAngle = 0;              // 根据电感偏移量计算出的临时打角值
sint16 LastAngle = 0;              // 记忆冲出赛道前的有效偏移方向

sint16 Lleft1 = 0, Lleft2 = 0, Lright2 = 0, Lright1 = 0;   // 电感偏移量
sint16 LnowADC[6];                 // 电感当前 ADC 数值

sint16 ad_max[6] = {2500, 2300, 999, 999, 2500, 2500}; // 新板子放到赛道标定最大值,会被程序刷新
sint16 ad_min[6] = {120, 120, 999, 999, 120, 120}; // 新板子据需要标定最小值,会被程序刷新

uint8 CircleNumber = 1;      // 入环次数，0 结束；默认 1 次 ;环的个数一般在比赛前测试赛道时就知道了
uint8 TangentPoint = 1;      // 切点判断    0 切点结束；默认 1 可以入环，读取脉冲为入环准备
uint8 EnterCircle = 0;       // 允许进环   默认 0 不可进环；1 可以进环
uint8 OutCircle = 0;         // 允许出环    默认 0 不可出环；1 可以出环
uint8 LeftRightCircle = 0; // 左侧环还是右侧环 默认 0 原始；1 左环；2 右环

sint32 TangentPointpulse = 0; // 脉冲记忆临时变量 1
sint32 EnterCirclePulse = 0;   // 脉冲记忆临时变量 2
sint32 OutCirclePulse = 0;      // 脉冲记忆临时变量 3
sint32 EnterCircleOKPulse = 0;// 脉冲记忆临时变量 4
```

```
sint16 LowSpeed = 0;                // 圆环/十字口减速

uint16 MagneticField = 0;       // 磁场强度       magnetic field intensity,判断圆环是否出现

sint16 OffsetDelta = 0;

int Flag_Round_zuo = 0,Flag_Round_you = 0,zuo_flag=0,you_flag=0,you_number_flag,zuo_number_flag
,zuo_ru_number,you_ru_number;
extern int y_duandian_you,y_duandian_zuo;



/*************************************************************************
 *  函数名称：void InductorInit (void)
 *  功能说明：四个电感 ADC 初始化函数；
 *  参数说明：无
 *  函数返回：无
 *  备      注：
 *************************************************************************/
void InductorInit (void)
{
    ADC_InitConfig(ADC0, 100000); // 初始化,采样率 100kHz
    ADC_InitConfig(ADC1, 100000); // 初始化
    ADC_InitConfig(ADC2, 100000); // 初始化
    ADC_InitConfig(ADC3, 100000); // 初始化
}
/*************************************************************************
 *  函数名称：void InductorNormal(void)
 *  功能说明：采集电感电压并归一化；
 *  参数说明：无
 *  函数返回：无
 *  备      注：    注意要先标定运放的放大倍数，尽量四个一致或者接近
 *************************************************************************/
```

```
void InductorNormal (void)
{
    LnowADC[0] = ADC_Read(ADC0);   // 左侧第 1 个电感，与赛道夹角约 30 度，采集各个电感的 AD 值
    LnowADC[1] = ADC_Read(ADC1);   // 左侧第 2 个电感，垂直赛道，
    LnowADC[4] = ADC_Read(ADC2);   // 右侧第 2 个电感，垂直赛道，
    LnowADC[5] = ADC_Read(ADC3);   // 右侧第 1 个电感，与赛道夹角约 30 度
    BatVolt          = ADC_Read(ADC7);   // 刷新电池电压
    if (LnowADC[0] < ad_min[0])
        ad_min[0] = LnowADC[0];        // 刷新最小值
    else if (LnowADC[0] > ad_max[0])
        ad_max[0] = LnowADC[0];        // 刷新最大值
    if (LnowADC[1] < ad_min[1])
        ad_min[1] = LnowADC[1];
    else if (LnowADC[1] > ad_max[1])
        ad_max[1] = LnowADC[1];
    if (LnowADC[4] < ad_min[4])
        ad_min[4] = LnowADC[4];
    else if (LnowADC[4] > ad_max[4])
        ad_max[4] = LnowADC[4];
    if (LnowADC[5] < ad_min[5])
        ad_min[5] = LnowADC[5];
    else if (LnowADC[5] > ad_max[5])
        ad_max[5] = LnowADC[5];

    Lleft1 = (LnowADC[0] - ad_min[0]) * 100 / (ad_max[0] - ad_min[0]);        // 各偏移量归一化到 0--100 以内
    Lleft2 = (LnowADC[1] - ad_min[1]) * 100 / (ad_max[1] - ad_min[1]);
    Lright1 = (LnowADC[4] - ad_min[4]) * 100 / (ad_max[4] - ad_min[4]);
    Lright2 = (LnowADC[5] - ad_min[5]) * 100 / (ad_max[5] - ad_min[5]);

    MagneticField = Lleft1 + Lleft2 + Lright2 + Lright1;// 磁场整体强度
```

```
                                        /*********************************以下为环岛判断**************************************/
    //以下判断条件不可直接使用，一定要根据环岛处图像和电感值特点进行调整，反复调试修改环岛通过率可以接近 100%
                //右环岛

if((Lleft1>2000)&&(Lleft1<2000)&&(Lleft1>200)&&Lleft1>1000&&Lleft1>600&&((Lleft1+Lleft2)>4000)//电磁条件
                  &&!Flag_Round_zuo&&y_duandian_zuo>28&&y_duandian_you>40)
//光电条件
          {
                you_number_flag++;
                if(you_number_flag>=1)
                {
                Flag_Round_you=1;
    //            led (LED0,LED_ON);
                }
          }
//                                                                                    else
if((g_ValueOfAD[2]>1800)&&(g_ValueOfAD[3]<500)&&g_ValueOfAD[1]>0&&g_ValueOfAD[0]>3000                    //电磁条件
//                          &&!Flag_Round_zuo&&y_duandian_zuo>20&&y_duandian_you>20)
//光电条件
//          {
//                you_number_flag++;
//                if(you_number_flag>=1)
//                {
//                Flag_Round_you=1;
//    //            led (LED0,LED_ON);
//                }
//          }
```

```
            //左环岛

if((Lleft1>2000)&&(Lleft1<2000)&&(Lleft1>200)&&Lleft1>1000&&Lleft1>600&&((Lleft1+
Lleft2)>4000)//电磁条件
            &&!Flag_Round_zuo&&y_duandian_zuo>28&&y_duandian_you>40)
//光电条件
        {

                Flag_Round_zuo=1;
    //            led (LED2,LED_ON);

        }
//                                                                          else
if((g_ValueOfAD[3]>1800)&&(g_ValueOfAD[2]<500)&&(g_ValueOfAD[0]>0)&&(g_ValueO
fAD[1]>3000)            //电磁条件
//                      &&!Flag_Round_you&&y_duandian_zuo>20&&y_duandian_you>20)
//光电条件
//        {
//
//                Flag_Round_zuo=1;
//    //            led (LED2,LED_ON);
//
//        }


 }

/*****************************************************************************
 *  函数名称：void CircleDetect void
 *  功能说明：识别并进入圆环的个数；
 *  参数说明：无
 *  函数返回：无
 *  备      注：
```

```
**********************************************************************/
void CircleDetect (void)
{
    if (CircleNumber) //  入环次数，0 结束；默认 2 次
    {
        //  进环切点区域判断
        if (MagneticField > 220) //  直道进入，此值不宜太大，容易丢失切点
        {
            if (TangentPoint)
            {
                TangentPointpulse = RAllPulse; //  读取当前脉冲数值
                TangentPoint = 0;                    //  禁止再次读取当前脉冲数值
            }
            else if (LeftRightCircle == 0)
            {                                                //  如果前面设置为 220，此处应该大
点儿，距离拉长一些，太大就走过切点了也不行！
                if (RAllPulse > TangentPointpulse + 3500)     //  进入切点后再前进 3000
脉冲，大约 50cm，
                {
                    EnterCircle = 1;          //  通过切点区域，可以入环
                }
            }
        }

        //  。约 1.2 米外进环无效，则需要重新识别切点
        if ((LeftRightCircle == 0) && (RAllPulse > TangentPointpulse + 8000)) //  约 1.2 米
外进环无效
        {
            EnterCircle = 0;     //  约 1.2 米外进环无效
            TangentPoint = 1;    //  重新识别切点
        }
        if ((EnterCircle) && (MagneticField > 260)) //  约 1.2 米内再次发现强磁场，入环
        {
```

```
            LowSpeed = 500;       // 减速
            // 。左侧入环处理
            if (Lleft1 + Lleft2 > Lright2 + Lright1)        // 左边入环，存在误判！！！
            {
                LeftRightCircle = 1;   // 左侧环为 1
                EnterCircle = 0;          // 入环后禁止再次入环
                EnterCirclePulse = RAllPulse;
                ServoCtrl(Servo_Left_Max);         // 舵机 PWM 输出，转向舵机打角控
制
                while (RAllPulse < EnterCirclePulse + 2800)
                {
                    delayms(1);           // 半打角度前进 1200 脉冲，约 20cm，龙邱 512
带方向编码器 1 米 5790 个脉冲
                }
            }
            else // 右侧入环处理
            {
                LeftRightCircle = 2;   // 右侧环为 2
                EnterCircle = 0;   // 入环后禁止再次入环
                EnterCirclePulse = RAllPulse;
                ServoCtrl(Servo_Right_Min);             // 舵机 PWM 输出，转向舵机打角
控制
                while (RAllPulse < EnterCirclePulse + 1800) // 用的是右侧的编码器，实
际走的距离近一点儿
                {
                    delayms(1);     // 半打角度前进 1200 脉冲，约 20cm，龙邱 512 带
方向编码器 1 米 5790 个脉冲
                }
            }
            EnterCircleOKPulse = RAllPulse; // 出环用
        }
        // 出环处理
        if ((LeftRightCircle > 0) && (RAllPulse > EnterCircleOKPulse + 3000))
```

```
        {
            EnterCircleOKPulse = 10000000; //禁止再次出环使能
            OutCircle = 1;   // 进环后可以出环
        }
        if ((OutCircle) && (MagneticField > 220)) // 入环标志为真才能入环
        {
            LowSpeed = 500;              // 减速
            // 左侧出环处理
            if (LeftRightCircle == 1)     //左边入环
            {
                //OutCircle = 0;       // 入环后禁止再次入环
                OutCirclePulse = RAllPulse;
                ServoCtrl(Servo_Left_Max);         // 舵机 PWM 输出，转向舵机打角控
制
                while (RAllPulse < OutCirclePulse + 2800)
                {
                    delayms(1); // 半打角度前进 1200 脉冲，约 20cm，龙邱 512 带方
向编码器 1 米 5790 个脉冲
                }
                OutCirclePulse = RAllPulse;
                ServoCtrl(Servo_Center_Mid-Servo_Delta*2/3);         // 舵机 PWM 输出，
反向打角
                while (RAllPulse < OutCirclePulse + 700)
                {
                    delayms(1);              // 半打角度前进 600 脉冲，约 10cm，龙邱
512 带方向编码器 1 米 5790 个脉冲
                }
                CircleNumber--;              // 环计数
                TangentPoint = 1;           // 切点判断     0 切点结束；默认 1 可以入环，
读取脉冲为入环准备
                EnterCircle = 0;            // 允许进环   默认 0 不可进环；1 可以进环
                OutCircle = 0;               // 允许出环     默认 0 不可出环；1 可以出环
                LeftRightCircle = 0;       // 左侧环还是右侧环  默认 0 原始；1 左环；2
```

```
右环
                LowSpeed = 0;                   // 恢复速度
                Reed_Init();                    // 干簧管 GPIO 及中断初始化函数,为停车入
库做准备
            }
            // 右侧出环处理
            else if (LeftRightCircle == 2)   //右边入环
            {
                //OutCircle = 0;        // 入环后禁止再次入环
                OutCirclePulse = RAllPulse;
                ServoCtrl(Servo_Right_Min);        // 舵机 PWM 输出，转向舵机打角控
制
                while (RAllPulse < OutCirclePulse + 2500)
                {
                    delayms(1);    // 半打角度前进 1200 脉冲，约 20cm，龙邱 512 带
方向编码器 1 米 5790 个脉冲
                }
                OutCirclePulse = RAllPulse;
                ServoCtrl(Servo_Center_Mid+Servo_Delta*2/3);        // 舵机 PWM 输出，
反向打角
                while (RAllPulse < OutCirclePulse + 1400)
                {
                    delayms(1);              // 半打角度前进 600 脉冲，约 10cm，龙邱
512 带方向编码器 1 米 5790 个脉冲
                }
                CircleNumber--;             // 环计数
                TangentPoint = 1;          // 切点判断    0 切点结束；默认 1 可以入环，
读取脉冲为入环准备
                EnterCircle = 0;           // 允许进环   默认 0 不可进环；1 可以进环
                OutCircle = 0;              // 允许出环    默认 0 不可出环；1 可以出环
                LeftRightCircle = 0;       // 左侧环还是右侧环 默认 0 原始；1 左环；2
右环
                LowSpeed = 0;                   // 恢复速度
```

```
                    Reed_Init();                    // 干簧管 GPIO 及中断初始化函数,为停车入
库做准备
                }
            }
        }
}

/*************************************************************************
 *  函数名称：void TFT_Show_EleMag_Info(void)
 *  功能说明：显示各种所需信息
 *  参数说明：无
 *  函数返回：无
 *  备      注：
 *************************************************************************/
void TFT_Show_EleMag_Info(void)
{
    char txt[16] = "X:";

    sint16 mps = 0, dmm = 0;       // 速度：m/s,毫米数值
    sint16 pulse100 = 0;
    uint16 bat=0;

    dmm = (sint16) (RAllPulse * 100 / 579);                 // 龙邱 512 带方向编码器 1 米 5790
个脉冲，数值太大，除以 100
    pulse100 = (sint16) (RAllPulse / 100);
    sprintf(txt, "AP:%05d00", pulse100);                    //
    TFTSPI_P8X16Str(3, 1, txt, u16RED, u16BLACK);       // 显示赛道偏差参数

    NowTime = (STM_GetNowUs(STM0) - NowTime) / 1000;   // 获取 STM0 当前时间，
得到毫秒
    mps = (sint16) (dmm / (NowTime / 1000));                // 计算速度 mm/s
    // 调试信息
    sprintf(txt, "%04d %04d %04d ", TempAngle, ECPULSE1, ECPULSE2);     // 显示舵机
```

```
角度数值，电机占空比数值，编码器数值
    TFTSPI_P8X16Str(1, 0, txt, u16WHITE, u16BLACK);          // 字符串显示
    //显示各电感归一化后的偏移量  当前各电感电压值 各电感开机后历史最小值 各电感开机后历史最大值
    sprintf(txt, "0:%04d %04d %04d ", Lleft1, LnowADC[0], ad_max[0]);
    TFTSPI_P8X16Str(0, 2, txt, u16WHITE, u16BLACK);
    sprintf(txt, "1:%04d %04d %04d ", Lleft2, LnowADC[1], ad_max[1]);
    TFTSPI_P8X16Str(0, 3, txt, u16WHITE, u16BLACK);
    sprintf(txt, "2:%04d %04d %04d ", Lright2, LnowADC[4], ad_max[4]);
    TFTSPI_P8X16Str(0, 4, txt, u16WHITE, u16BLACK);
    sprintf(txt, "3:%04d %04d %04d ", Lright1, LnowADC[5], ad_max[5]);
    TFTSPI_P8X16Str(0, 5, txt, u16WHITE, u16BLACK);

    sprintf(txt, "Ring num: %d ", CircleNumber);
    TFTSPI_P8X16Str(2, 6, txt, u16GREEN, u16BLACK);
    sprintf(txt, "M:%03d Q:%d J:%d ", MagneticField, TangentPoint, EnterCircle);
    TFTSPI_P8X16Str(0, 7, txt, u16WHITE, u16BLACK);
    if (LeftRightCircle == 1)
        TFTSPI_P8X16Str(0, 8, "Left Ring ", u16WHITE, u16BLACK);
    else if (LeftRightCircle == 2)
        TFTSPI_P8X16Str(0, 8, "Right Ring", u16WHITE, u16BLACK);
    else
        TFTSPI_P8X16Str(0, 8, "No Ring      ", u16WHITE, u16BLACK);

    bat = BatVolt * 11 / 25;    // x/4095*3.3*100*5.7
    sprintf(txt, "B:%d.%02dV %d.%02dm/s", bat / 100, bat % 100, mps / 1000, (mps / 10) % 100);    // *3.3/4095*3
    TFTSPI_P8X16Str(0, 9, txt, u16PURPLE, u16BLACK);     // 字符串显示
}
/*********************************************************************
 *  函数名称：void ElectroMagneticCar(void)
 *  功能说明：智能车双电机差速控制
 *  参数说明：无
```

```
 *  函数返回：无
 *  备      注：驱动 2 个电机
 ***********************************************************************/
void ElectroMagneticCar (void)
{
    sint16 bat=0;

    CircleNumber = 1;    // 入环次数，0 结束；默认 1 次
    TangentPoint = 1;    // 切点判断    0 切点结束；默认 1 可以入环，读取脉冲为入环准备
    EnterCircle = 0;     // 允许进环   默认 0 不可进环；1 可以进环
    OutCircle = 0;       // 允许出环    默认 0 不可出环；1 可以出环
    LeftRightCircle = 0;// 左侧环还是右侧环 默认 0 原始；1 左环；2 右环
    LowSpeed = 0;        // 速度差

    // 切记 CPU0,CPU1...不可以同时开启屏幕显示，否则冲突不显示
    mutexCpu0TFTIsOk=0;                 // CPU1： 0 占用/1 释放 TFT
    CircleNumber = SetCircleNum();  // 设置需要进入圆环的个数；

    // 。根据需要设置出入库，出库是固定执行，
    // 。入库需要干簧管和外部中断配合实现
    // 。本例程中，干簧管在通过圆环后开启，不会出现起跑触发的可能性
    OutInGarage(OUT_GARAGE,ReadOutInGarageMode()); // 测试出库，拨码在上左侧出入库，反之右侧出入库
    //OutInGarage(IN_GARAGE,ReadOutInGarageMode());  // 测试入库

    TFTSPI_CLS(u16BLACK);               // 清屏
    // 切记 CPU0,CPU1...不可以同时开启屏幕显示，否则冲突不显示
    mutexCpu0TFTIsOk=1;                 // CPU1： 0 占用/1 释放 TFT

    RAllPulse = 0;                      // 全局变量，脉冲计数总数
    NowTime = STM_GetNowUs(STM0);   // 获取 STM0 当前时间
```

```
    while (1)
    {
        InductorNormal();                  // 采集电感电压并并归一化；
        if (MagneticField > 220)       // 直道进入，此值不宜太大，容易丢失切点
        {
            LowSpeed = 500;                // 减速
        }
        else if (MagneticField < 200)
        {
            LowSpeed = 0; // 恢复速度
        }
        CircleDetect();                    // 识别并进入圆环的个数；

        ServoDuty = TempAngle;
        ServoCtrl(ServoDuty);              // 舵机 PWM 输出，转向舵机打角控制
        //
        OffsetDelta = (Lleft2 - Lright2);   // 直道偏差
        bat=(BatVolt * 11 / 25-750)*Kbat;
        MotorDuty1 = MtTargetDuty-bat + ECPULSE1 * Kencoder - OffsetDelta * Koffset -
LowSpeed;   // 有差速控制，右转偏差为负，左侧加速
        MotorDuty2 = MtTargetDuty-bat - ECPULSE2 * Kencoder + OffsetDelta * Koffset -
LowSpeed;   // 有差速控制，左转偏差为正，右侧加速

        if (MagneticField < 30)        // 判断是否冲出赛道
        {
            MotorCtrl(0, 0);               // 冲出赛道停车
            delayms(200);
        }
        else
        {
            MotorCtrl(MotorDuty1, MotorDuty2);     // 四轮双电机驱动
            // MotorCtrl(MtTargetDuty-TempAngle*8/5, MtTargetDuty+TempAngle*8/5);//
三轮车，无舵机
```

```
            }

            if(Game_Over)
            {
                OutInGarage(IN_GARAGE, ReadOutInGarageMode());
            }
        } // WHILE(1)
} // MAIN()

float constrain_float(float amt, float low, float high)
{
    return ((amt)<(low)?(low):((amt)>(high)?(high):(amt)));
}

// pid 参数初始化函数
void PidInit(pid_param_t * pid)
{
    pid->kp             = 0;
    pid->ki             = 0;
    pid->kd             = 0;
    pid->imax           = 0;
    pid->out_p          = 0;
    pid->out_i          = 0;
    pid->out_d          = 0;
    pid->out            = 0;
    pid->integrator= 0;
    pid->last_error= 0;
    pid->last_derivative    = 0;
    pid->last_t         = 0;
}

/*************************************************************************
 *  函数名称：float constrain_float(float amt, float low, float high)
```

```
 *   功能说明：pid 位置式控制器输出
 *   参数说明：
  * @param      pid        pid 参数
  * @param      error     pid 输入误差
 *   函数返回：PID 输出结果
 *   备      注：
 ************************************************************************/
float PidLocCtrl(pid_param_t * pid, float error)
{
    /* 累积误差 */
    pid->integrator += error;

    /* 误差限幅 */
    constrain_float(pid->integrator, -pid->imax, pid->imax);



    pid->out_p = pid->kp * error;
    pid->out_i = pid->ki * pid->integrator;
    pid->out_d = pid->kd * (error - pid->last_error);

    pid->last_error = error;

    pid->out = pid->out_p + pid->out_i + pid->out_d;

    return pid->out;
}
/************************************************************************
 *   函数名称：float constrain_float(float amt, float low, float high)
 *   功能说明：pid 增量式控制器输出
 *   参数说明：
  * @param      pid        pid 参数
  * @param      error     pid 输入误差
 *   函数返回：PID 输出结果    注意输出结果已经包涵了上次结果
```

```
 *  备      注：
 ***************************************************************************/
float PidIncCtrl(pid_param_t * pid, float error)
{

    pid->out_p = pid->kp * (error - pid->last_error);
    pid->out_i = pid->ki * error;
    pid->out_d = pid->kd * ((error - pid->last_error) - pid->last_derivative);

    pid->last_derivative = error - pid->last_error;
    pid->last_error = error;

    pid->out += pid->out_p + pid->out_i + pid->out_d;

    return pid->out;
}

#ifndef SRC_APPSW_TRICORE_USER_LQ_PID_H_
#define SRC_APPSW_TRICORE_USER_LQ_PID_H_

#include <stdint.h>

typedef struct
{
    float                   kp;             //P
    float                   ki;             //I
    float                   kd;             //D
    float                   imax;           //积分限幅

    float                   out_p;  //KP 输出
    float                   out_i;  //KI 输出
    float                   out_d;  //KD 输出
    float                   out;    //pid 输出
```

```
    float                integrator; //< 积分值
    float                last_error; //< 上次误差
    float                last_derivative;//< 上次误差与上上次误差之差
    unsigned long        last_t;      //< 上次时间
}pid_param_t;


void PidInit(pid_param_t * pid);

float constrain_float(float amt, float low, float high);

float PidLocCtrl(pid_param_t * pid, float error);

float PidIncCtrl(pid_param_t * pid, float error);

#endif /* SRC_APPSW_TRICORE_USER_LQ_PID_H_ */
```

致　谢

在本篇论文完成之际，我要向所有支持和帮助过我的人表示最衷心的感谢。

首先，我要感谢我的导师温国强副教授，感谢他在论文的选题、设计、实验和撰写过程中给予的悉心指导和支持，从选题、论文结构及内容修改等环节导师始终认真负责，帮助我解决论文写作过程中遇到的各种难题，为我提供宝贵的修改意见，使我顺利完成了论文的写作工作。导师知识渊博、治学严谨，都深深地影响着我，激励我不断地进取。他的专业知识和严谨治学精神深深影响了我。

同时，我也要感谢实验室的所有老师和同学，他们的热情帮助和启发让我受益匪浅。其次，我要感谢我的家人和朋友，感谢他们一直以来的关心和支持，他们的理解和鼓励让我在学业上更加坚定和努力。

最后，我要感谢所有对本篇论文提出宝贵意见和建议的专家和评审，感谢他们的认真审阅和指导，帮助我不断完善和提高论文的质量。再次感谢所有支持和帮助过我的人，是你们的支持和鼓励让我完成了本篇论文。致以最诚挚的敬意！