学　号 **04041237**

密　级 _____

哈尔滨工程大学本科生毕业论文

智能车模拟自动驾驶软件系统设计

院　　（系）：　自动化学院

专　　业：　自动化

学 生 姓 名：　张望舒

指 导 教 师：　莫宏伟　教授

2008 年 6 月

哈尔滨工程大学本科生毕业论文

智能车模拟自动驾驶软件系统设计

院　　（系）：　自动化学院

专　　业：　自动化

学　　号：　04041237

学 生 姓 名：　张望舒

指 导 教 师：　莫宏伟　教授

2008 年 6 月

摘　　要

本课题属于智能交通系统(Intelligent Transportation System, ITS)领域的研究范畴。作为该领域的重要组成部分，智能车模拟自动驾驶技术的研究和发展为改善交通环境状况，提高车辆安全性与可靠性，减少驾驶员人为因素造成的交通事故等开辟了广阔的前景。

本文分四部分来设计智能车系统：视频处理及显示部分、图像处理部分、目标识别部分与控制部分。

在视频处理及显示部分，本文应用 VFW 软件包对视频进行实时捕捉和显示；在道路图像处理部分，本文对摄像头采集到的图像相继进行了灰度化处理、中值滤波、Sobel 边缘检测、阈值分割和数学形态学的处理；在目标识别部分，本文分别运用了基于道路形状的算法和改进的模板匹配法进行道路边缘识别与交通灯识别；在控制部分，本文介绍了智能车的控制方法和硬件部分的设计与实现。最后本文针对设计过程中的不足进行总结及提出自己的建议与意见，并展望智能车模拟自动驾驶技术的发展前景。

关键词：智能车模拟自动驾驶技术；视频捕捉；图像处理；目标识别

ABSTRACT

This thesis belongs to the research field of Intelligent Transportation System (ITS). As an important section of this field, the research and development of the Intelligent Automatic Vehicle Driving Simulation Technique (IAVDST), paves a broad prospect for the improvement of transportation envirionment, the enhancement of vehicles' safety and reliability, as well as the reduction of traffic accidents caused by human factors.

In this thesis, the Intelligent Vehicle System is designed in four parts, i.e. the video processing and display part, the image processing part, the object recognition part and the controlling part.

In the first part, the VFW software kit is applied to real-timely capture and display the video; in the second part, graying processing, median filtering, Sobel edge detection, threshold segmentation and mathematical morphology are applied to process the image collected by the camera; in the third part, the road shape based algorithm and the improved template matching method are applied respectively for the road edge recognition and traffic light recognition; in the fourth part, the controlling method and the hardware design and realization of the imtelligent vehicle are introduced in this thesis. Finally, refering to some deficiency in the designing process, some suggestions are proposed, and the prospect is also viewed for the IAVDST.

Keywords: Intelligent Automatic Vehicle Driving Simulation Technique; video capture; image processing; object recognition

目　　录

第 1 章 绪论

1.1 选题背景

2007 年 11 月 3 日，无人驾驶车比赛(Urban Challenge)在美国加利福尼亚州沙漠中一座空旷的城镇中举行，由美国 11 所大学设计的无人驾驶汽车展开速度与设计上的较量。参赛车辆在完全依靠自动驾驶的情况下，在各种路况下穿梭，它们在红灯时自动停车，绿灯时起步行走，在行驶过程中完全由电脑导航装置控制，由雷达、激光传感器和摄像机捕捉路面状况，并根据程序作出制动、转弯等反应。利用最先进的软件技术，它可以分析车辆周围的交通状况，并做出正确的驾驶操作[1]。

智能车辆(Intelligent Vehicle, IV)技术的研究，可以追溯到 20 世纪 50 年代初美国 Barrette Electronics 公司研制的世界上第一台自动引导车辆(Automated Guided Vehicle, AGV)，它实际上是低速智能车辆研究领域的一台移动式机器人[1]。而在高速智能车辆研究领域，早在 1939－1940 年纽约世界博览会上，美国通用汽车公司(GM)展示的 Futurama 概念车就提出汽车自动驾驶的概念。从 20 世纪 50 年代后期到 60 年代中期，美国、德国、英国、日本等国家都开展了高速车辆自动驾驶的研究，其自动驾驶系统普遍采用在行车线埋设引导电缆来实现车辆转向控制。日本于 1977 年首先开发出了机器视觉导航自动驾驶汽车的样车，20 世纪 80 年代前期，美国开发出了军用自动驾驶越野车。从 20 世纪 80 年代后期开始，为解决日益突出的交通事故、交通堵塞、交通环境污染、能源消耗等问题，各发达国家开始投入了大量的人力、物力研究实施智能交通系统(ITS)。ITS 的基本指导思想是将信息技术、数据通讯传输技术、电子控制技术及计算机处理技术等综合地运用于整个公路交通体系，从而有效地提高交通安全、优化交通运输资源、减少能源消耗、降低环境污染等，实现车辆与道路综合交通系统的智能化。交通系统智能化离不开智能车辆技术的支持，而 ITS 的兴起极大地促进了智能车辆技术水平的提高，智能车辆作为 ITS 的重要组成部分得以被系统地研究开发。这一阶段，智能车

辆研究项目多包含在美国的 IVHS、欧洲的 Prometheus、日本的 ITS 等总体项目中，智能车辆相关技术也相继取得了突破性的发展，如德国的 VaMoRs-P 车辆系统、美国的 NavLab 系统、意大利的 ARGO 系统等[2]。上述无人驾驶汽车挑战赛即是由美国国防部先进技术研究署(DARPA)投入巨额奖金举办的一项高科技赛事。

1.2 本课题研究的目的及意义

汽车是人类发展过程中的重要产物，它为现代社会的发展和人类生活条件的改善作出了巨大贡献。但是随着城市化的发展及汽车的日益普及，交通环境日趋恶劣，交通拥挤加剧，交通事故频发，交通问题已经成为全球范围内人们普遍关注的社会问题。因此，提高汽车安全性能，减少道路交通事故，改善交通环境就成为关系到人类社会进步与可持续发展的重要研究课题之一。

近年来，为解决交通问题世界各国都竞相开展智能车路系统(Intelligent Vehicle Highway System, IVHS)和智能交通系统(Intelligent Transportation System, ITS)等领域的研究。在公路交通中，智能车辆作为其中的重要组成部分，相关研究越来越受到人们的关注和重视。智能车辆的发展为改善交通环境状况，提高车辆安全性与可靠性，减少驾驶员人为因素造成的交通事故等开辟了广阔的前景。

1.3 智能车辆研究现状

1.3.1 国外研究现状

从 1987 年到 1994 年，在欧洲开展了“普罗米修斯”(Prometheus－Programme for a European traffic of highest efficiency and unprecedented safety) EUREKA 研究项目。该项目中颇具代表性的是戴姆勒－奔驰公司研制的 VITAⅡ试验车，于 1994 年 10 月在巴黎附近的一号高速公路上进行了车辆导航试验，在长达几千公里的普通三车道路段中采用了驾驶员辅助驾驶和车辆

自主驾驶相结合的导航方法。

德国联邦国防大学(UBM)从 20 世纪 80 年代初期就开始进行智能车辆自主导航的研究，其合作伙伴是德国戴姆勒－奔驰汽车公司。其中最具代表性的是由一辆豪华型奔驰 500SEL 改装而成的 VaMoRs-P 试验车。VaMoRs-P 试验车在高速公路和普通标准公路上进行了大量试验，试验内容包括跟踪车道线，躲避障碍以及自动超车等。

美国卡内基梅隆大学(CMU)机器人研究所在智能车辆研究领域也开展了深入的研究，研制开发了 NavLab 系列试验车。其中最具代表性的是根据运动跑车(Pontiac Trans Sport)改装而成的 NavLab-5 试验车，由美国著名的 Delco Electronics 公司捐资赞助。该试验车进行了横穿美国大陆的长途自主驾驶试验，其中车辆纵向控制由驾驶员完成，而车辆的横向导航控制则完全实现了自动控制。

意大利 Parma 大学信息工程系在智能车辆领域进行了大量研究，研制的 ARGO 试验车，其显著特点是采用了商用低成本计算机系统和传感器系统。在 1998 年意大利汽车百年行活动中，ARGO 试验车由 GOLD 系统驾驶进行了 2000km 的道路试验，在试验中其自动驾驶里程达到总行程的 94％。

另外，德国研究与技术部门与德国大众汽车公司(VolksWagen)合作研制了 Caravelle 试验车，车体采用大众公司的 Caravell 旅行车改装而成，目前正在开展隔离试验道路上计算机视觉与其它传感器信息融合的完全自主驾驶试验研究。法国帕斯卡大学自动化与电子材料实验室与法国 D.R.A.S 雪铁龙(Citroen)技术中心合作，联合研制出一个功能简单却颇具特色的辅助导航 Peugeot 试验车。该研究由 P.S.A 标志(Peugeot)雪铁龙汽车公司提供资助。该试验车的一个突出特点是硬件配置轻型化，整个系统的运算处理部分都已集成在一块数字信号处理卡上，因此对试验车几乎无需作任何改装。Peugeot 试验车已经在高速公路上进行了几百公里不同路况的行车试验，其系统具有良好的适应性。

在亚洲，日本丰田公司(TOYOTA)也研制出由丰田轿车改造而成的智能

试验车，并进行了相关的导航试验。另外，日本专利局(JPO－MITI)与日产汽车公司(Nissan)、富士通公司(Fujitsu)合作开展 PVS(Personal Vehicle System)研究项目,其自动驾驶汽车系统在夜间和雨天环境下也能实现自动驾驶。

1.3.2 国内研究现状

随着我国改革开放的不断发展和科技水平的日益提高，从八五期间也开始了智能车辆方面的研究。但由于起步较晚，以及经济条件的制约，我国在智能车辆研究领域与发达国家仍有一定的差距，目前开展这方面研究工作的单位主要有国防科技大学、清华大学、吉林大学、北京理工大学、中科院沈阳自动化研究所等。

国防科技大学机电工程与自动化学院自 20 世纪 80 年代起开始进行无人驾驶智能车辆技术研究，先后研制出四代无人驾驶汽车。第四代自主无人驾驶汽车于 2000 年 6 月在长沙市环城高速公路上进行了试验。从 1993 年开始，由南京理工大学、北京理工大学、浙江大学、国防科技大学、清华大学等多所院校联合研制开发“地面军用智能机器人”项目，选用国产跃进客货车改制而成的智能车上集成了二维彩色摄像机、三维激光雷达、陀螺惯导定位、超声等传感器。其体系结构以水平式结构为主，采用传统的“感知—建模—规划—执行”算法，其直线跟踪速度达到 20km/h，避障速度达到 5～10km/h。

在国防科工委和国家 863 计划的资助下，清华大学计算机系智能技术与系统国家重点试验室自 1988 年开始研制 THMR(Tsinghua Mobile Robot)系列移动机器人系统。其中 THMR-V 系统是清华大学计算机系目前正在研制的新一代智能移动机器人，兼有面向高速公路和一般道路的功能。车体采用道奇 7 座厢式车改装，装备有彩色摄像机、GPS、磁罗盘光码盘定位系统、激光测距仪 LMS220 等。它的体系结构以垂直式为主，采用多层次“感知—动作”行为控制及基于模糊控制的局部路径规划及导航控制。该智能车设计车速为 80km/h，一般道路为 20km/h。

1.4 研究任务和论文结构安排

1.4.1 研究任务

限于实际条件，本课题以自制的车模型为基础，在实验室条件下模拟简易的道路交通系统，主要研究数字图像处理及模式识别的相关算法。本设计的主要任务包括：智能车行驶过程中道路边缘的跟踪识别、红绿灯的检测，以及智能车模拟自动驾驶软件界面的设计。

1.4.2 论文结构安排

本论文的结构是这样安排的：

第一章主要介绍课题研究的背景、意义、国内外研究现状以及本设计要完成的主要内容。

第二章主要介绍智能车视觉系统的设计方法、摄像头的选择以及智能车前方视野视频的实时采集和显示方法。

第三章主要应用数字图像处理学的基本原理和算法对实时采集到的视频图像进行数字图像处理。

第四章主要介绍道路边缘和交通灯的目标识别算法。

第五章主要进行智能车系统综合设计和调试。

最后是结论，对本文所做的工作进行总结，在论文的最后附上参考文献和致谢。

第2章 智能车前方视野的视频实时采集

2.1 引言

视觉是人类观察世界、认识世界的重要功能手段，人类从外界获得的信息约有 75%来自视觉系统，特别是驾驶员驾驶需要的信息 90%来自视觉，如交通信号、交通图案、道路标识等均可以看作是环境对驾驶员的视觉通讯语言。于是，人们自然考虑应用计算机视觉来解释这些环境语言。从图像处理与模式识别发展起来的计算机视觉（也称机器视觉），能够利用图像和图像序列来识别和认知三维世界，使计算机实现人类视觉的某些功能。

视觉系统在智能车辆研究中主要起到环境探测和辨识的作用。与其它传感器相比，计算机视觉具有检测信息量大、适应范围广、智能化程度高等优点。目前计算机视觉已成为智能车辆环境信息获取的主要手段，对智能车辆视觉系统的研究已成为智能车辆研究领域广泛关注的热点之一，所以本章对智能小车的视觉系统进行了相关研究，为后续的图像处理与识别创造良好条件。

2.2 智能车视觉系统设计

2.2.1 总体设计要求

从理论上分析，要获得道路环境的三维信息，需要采用双目或多目立体视觉系统。但是，双目或多目立体视觉系统在实际应用中所需计算量很大，而智能车辆在较高速度下的图像处理频率比一般情况下更高，目前的微处理器计算能力还不能完全满足其实时性的要求，所以目前还不适合在较高速度下智能车辆视觉导航中应用。智能车辆视觉系统主要是获取道路平面的二维路径信息，而道路中的其它车辆和障碍物信息可以通过视觉系统、激光雷达测距仪及避障传感器系统进行信息融合得到，这就极大的提高了信息获取的可靠性，所以单目视觉系统仍然能够满足较高速度情况下视觉导航的要求。

实际上，世界范围内大多数智能车辆视觉导航系统都采用单目视觉来获取道路环境信息，如美国卡内基梅隆大学的 NabLab 视觉系统、德国的 Caravelle 视觉系统及法国的 Peugeot 视觉系统等。

在本设计的实验条件下，待检测的道路是理想化的结构化道路，检测信息量也较少，因此本智能车也采用单目视觉系统。接下来在视觉传感器的选择方面，在常用的 CCD 摄像机与 CMOS 摄像头之间做出选择，对于视频读取和图像处理来说，后者的性价比要高于前者，因此本设计采用普通视频聊天用的摄像头作为视觉传感器。

2.2.2 摄像头的选择

摄像头的选择主要包括摄像头焦距、视场角等参数的选择，以及与此相关的摄像头安装高度与摄像头俯仰角等参数的确定。相关参数选择时需要满足车辆预瞄距离的要求。

(1) 安装高度 h

摄像头的视野范围与摄像机的安装高度有关，在相同的俯仰角情况下摄像机安装高度增大，摄像头的视野范围也随之增大，车辆的预瞄距离也会相应的增加。所以为了扩大摄像头的视野范围，在安装时可以适当增加摄像头的安装高度。

(2) 摄像头的预瞄距离 d

本文定义摄像头的预瞄距离 d 为摄像头成像下边缘与地面交点到摄像头在地面上的垂直投影点之间的水平距离，如图 2.1 所示。车辆行驶时的预瞄距离 d 可由经验公式(2.1)估算：

$$d = v \cdot k_t \tag{2.1}$$

其中，v 是车辆行驶速度，可以通过车轮半径 r 和电机转速 n 估算出来，如公式(2.2)所示；k_t 是预瞄时间，一般取 0.3～1.5s。

$$v = 2\pi \cdot r \cdot \frac{n}{60} \tag{2.2}$$

(3) 摄像头水平视场角 β_0 与焦距 f

镜头的焦距与视场角是一一对应的，而且是矛盾的。即镜头焦距小，则视场角大，视野范围大，但距离远的物体分辨不很清楚；反之，焦距大，则视场角小，视野范围小，但距离远的物体分辨更清楚。摄像头视野应在车辆预瞄点处覆盖整个道路区域，在图 2.2 中 w 为道路宽度，则估算摄像头的水平视场角的依据是在道路预瞄点处摄像机的视野范围的宽度大小至少为道路宽度 w，由几何关系有：

$$\beta_0 = 2\arctan\frac{w}{2\cdot\sqrt{h^2+d^2}} \tag{2.3}$$

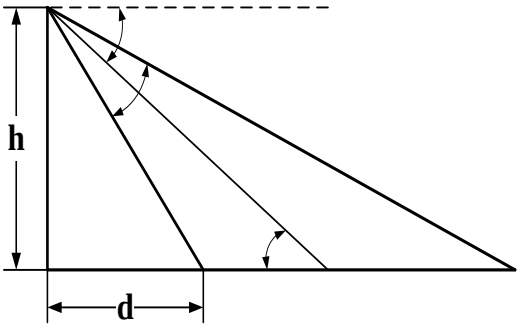

图 2.1 摄像头安装参数示意图

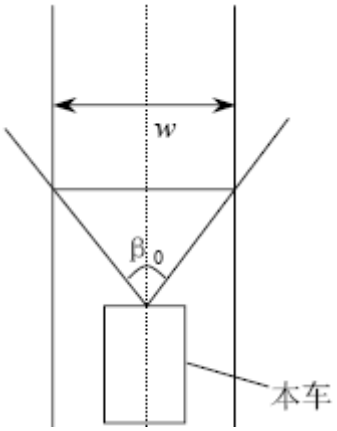

图 2.2 视场角估算

针对该智能车系统， 车辆的行驶速度为：

$$v = 2\pi\cdot r\cdot\frac{n}{60} = 2\pi\times 0.02\times\frac{60}{60} = 0.1256\,m/s$$

摄像头的预瞄距离 d 为：

$$d = v\cdot k_t = 0.1256\times 1.5 = 0.1884\,m$$

在这里道路的宽度为 $60\,cm$，即 $w = 0.6\,m$，则摄像头的水平视场角可以通过下式求出：

$$\beta_0 = 2\arctan\frac{w}{2\cdot\sqrt{h^2+d^2}} = 2\arctan\frac{0.6}{2\cdot\sqrt{0.15^2+0.1884^2}} = 102.4895^\circ$$

根据以上要求，选取中星微 301P 型摄像头，该摄像头的主要参数如表 2.1 所示：

表 2.1 中星微 301P 主要性能参数

基本参数	感光器件	CMOS
	传感器像素	35 万
	接口类型	USB 接口
技术功能	最大分辨率	320*240
	最大帧频	30 fps
	色彩位数	真彩色 24 位
	视场	55 ±度
	成像距离	50 毫米到无限远
	对焦方式/范围	6 毫米到无限远

2.3 视频捕捉软件包 VFW 的应用

VFW 是 Microsoft 公司 1992 年推出的关于数字视频的一个软件包，它能使应用程序数字化并播放从传统模拟视频元得到的视频剪辑。它是 Microsoft DirectX 中 Microsoft DirectShow API 里的一部分。该开发工具包含了开发视频应用程序所需的接口函数，使用这些函数可以从视频源中采集数字视频信号， 并将其存储到文件中或直接对视频缓存进行处理。VFW 的一个关键思想是播放时不需要专用硬件，为了解决数字视频数据量大的问题，需要对数据进行压缩。于是，它引进了 AVI 文件标准，该标准未规定如何对视频进行捕获、压缩和播放，仅规定视频和音频该如何存储在硬盘上，在 AVI 文件中交替存储视频帧和与之相匹配的音频数据。VFW 给程序员提供.VBX 和 AVICap 窗口类的高级编程工具，使程序员能通过发送消息或设置属性来捕获、播放和编辑视频剪辑。现在用户不必专门安装 VFW 了，Windows 95 本身包括了 Video for Windows1.1，当用户在安装 Windows 操作系统时，安装程序会自动安装配置视频所需的组件，如设备驱动程序、视频压缩程序等。

VFW 主要由以下 6 个模块组成：

(1) AVICAP.DLL：包含了执行视频捕获的函数，它给 AVI 文件 I/O 和视频、音频设备驱动程序提供了一个高级接口。

(2) MSVIDEO.DLL：用一套特殊的 DrawDib 函数来处理屏幕上的视频操

作。

(3) MCIAVI.DRV：此驱动程序包括对 VFW 的 MCI 命令的解释器。

(4) AVIFILE.DLL：支持由标准多媒体 I/O (MIMIO)函数提供的更高级的命令来访问 AVI 文件。

(5)压缩管理器(ICM)：管理用于视频压缩--解压缩的编解码器(CODEC)。

(6)音频压缩管理器(ACM)：提供与 ICM 相似的服务，不同的是它适于波形音频[5]。

视频数据的实时采集主要是通过调用 AVICAP.DLL 创建 AVICap 窗口类，由 AVICap 窗口类中的消息、宏函数、结构以及回调函数来完成。AVICap 在捕获视频方面具有明显的优势，它能直接访问视频缓冲区，不需要生成中间文件，实时性很高，它也可将数字视频保存到事先建好的文件中。实际应用表明，通过这种方法，提高了视频采集的效果和程序运行的效率，同时也减少了对硬件的依赖性，提高了程序的兼容性和移植性。

1、AVICap 窗口类简介

AVICap 支持实时的视频流捕捉和视频单帧捕捉。使用 AVICap 窗口类可创建具有一些基本功能的窗口，例如视频图像的预览、设置捕捉参数的对话框、音频、视频捕捉的独立控制等。AVICap 中的回调函数可使应用程序向用户提供有关捕捉的状态，包括进行的过程指示，以及任何可能产生的错误。开发人员可以设置一个标志用来指示在什么时候采集到音频，什么时候采集到视频。这样，应用程序可以直接使用数据而无需写入 AVI 文件中。

AVICap 窗口类提供了以下功能[3]：

(1) 单独控制音频、视频的采集；

(2) 采用 overlay（实时叠加）或 preview（预览）方式显示视频图像；

(3) 与 ICM 和 ACM 同时工作，将音频和视频数据直接压缩到应用程序中；

(4) 将音频、视频流直接压缩入 AVI 文件而不需要开发人员详细了解 AVI 文件格式的细节；

(5) 动态了解视频和音频的输入设备；

(6) 创建、保存和载入调色板；

(7) 将图像调色板拷贝到剪切板上；

(8) 控制 MCI 设备；

(9) 捕捉单帧图像并以 DIB 格式保存。

2、AVICap 窗口类的主要函数、宏简介

AVICap 提供给开发人员一整套函数，用这些函数可以实现许多视频捕捉程序所需的窗口管理；同时，在整个捕捉过程中仍然保留全部的控制。这些函数形式简单，采用基于消息的接口来获取硬件里的音频和视频信号，同时控制着视频流采集到磁盘的过程。

AVICap 的函数能够使开发人员以很少的投入来创建具有基本捕捉功能的采集程序，这些函数是高级的、经过优化的、为开发人员创建具有自己特性的应用程序留有很大的灵活性[4]。

下面是 AVICap 提供给开发人员编写捕捉程序的几个重要函数和宏。

(1) 创建捕捉窗口

```
HWND VFWAPI capCreateCaptureWindow(
LPCSTR lpszWindowName，// 捕捉窗口名字
DWORD dwStyle，// 捕捉窗口的风格
int x，// 窗口左上角 x 轴坐标
int y，// 窗口左上角 y 轴坐标
int nWidth，// 窗口的宽度
int nHeight，// 窗口的高度
HWND HWnd，// 父窗口句柄
Int nID// 捕捉窗口的 ID 号
);
```

如果该函数调用成功，则函数返回窗口的句柄，否则函数返回 NULL。

(2) 捕捉窗口与设备连接

BOOL capDriverConnect(

hwnd，// 捕捉窗口的句柄

iIndex// 设备驱动号

);

如果连接成功，返回 TRUE，否则函数返回 FALSE。

(3) 获取视频捕捉设备功能

BOOL capDriverGetCaps(

hwnd，// 捕捉窗口句柄

psCaps，// 指向一个用于存储返回值 CAPDRIVERCAPS 结构的指针

wSize// CAPDRIVERCAPS 结构占用的字节数

);

如果捕获窗口与捕获驱动连接成功，则返回 TURE，否则返回 FALSE。

(4) 设置捕捉设备的参数

BOOL capCaptureSetSetup(

hwnd，// 设置窗口句柄

psCapParms ， // 指向一个用于存储返回值的 CAPDRIVERCAPS 结构的指针

wSize// CAPDRIVERCAPS 结构占用的字节数

);

设置成功返回 TURE，否则返回 FALSE。

(5) 设置处理回调状态

BOOL capSetCallbackOnStatus(

hwnd，// 捕捉窗口句柄

fpProc// 指向状态回调函数的指针

);

(6) 设置错误处理

BOOL capSetCallbackOnError(

hwnd，// 捕捉窗口句柄

fpProc// 指向错误回调函数的指针

);

(7) 设置数据文件名

BOOL capFileSetCaptureFile(

hwnd，// 捕捉窗口句柄

szName// 指向字符串的指针，字符串内容为捕获文件名

);

设置一个由 szName 指向的字符串作为文件名，用于存储从捕捉窗口 hWnd 采集的视频图像数据，成功返回 TRUE，否则返回 FALSE。

(8) 开始捕捉视频

BOOL capCaptureSequence(

hwnd

);

触发程序开始捕捉视频图像并将其保存到数据文件。

(9) 视频源设置对话框

BOOL capDlgVideoSource(hwnd); // hwnd：捕捉窗口句柄

视频源设置对话框对于每一个捕捉驱动程序来说，是唯一的。而且，有些驱动程序不一定支持这一功能。应用程序可以通过检测 CAPDRIVERCAPS 结构的成员变量 fHasDlgVideoSource 来判断驱动程序是否支持这一功能。

(10) 视频格式设置对话框

BOOL capDlgVideoFormat(hwnd); // hwnd：捕捉窗口句柄

视频格式设置对话框对于每一个捕捉驱动程序来说，是唯一的。而且有些驱动程序不一定支持这一功能。应用程序可以通过检测 CAPDRIVERCAPS 结构的成员变量 fHasDlgVideoFormat 来判断驱动程序是否支持这一功能。

(11) 视频显示方式设置对话框

BOOL capDlgVideoDisplay(hwnd)； // hwnd：捕捉窗口句柄

视频格式设置对话框对于每一个捕捉驱动程序来说，是唯一的。而且有些驱动程序不一定支持这一功能。应用程序可以通过检测 CAPDRIVERCAPS 结构的成员变量 fHasDlgVideoDisplay 来判断驱动程序是否支持这一功能。

(12) 视频压缩设置对话框

BOOL capDlgVideoDisplay(hwnd)， // hwnd：捕捉窗口句柄

视频格式设置对话框允许用户在视频捕获期间选择不同的压缩器。

3、几个重要结构

编写视频捕捉程序往往要用到以下与视频捕捉相关的结构：

(1) CAPTUREPARMS：包括控制视频流捕捉过程的参数，这一结构被用来得到和设置影响捕捉速率、捕捉时的缓冲区数目、以及捕捉如何结束时的参数。

(2) CAPSTATUS：定义了捕捉窗口的当前状态，如：以象素为单位表示图像的高、宽、预览和重叠方式的标志量，尺寸缩放的标志量等。

因为捕捉窗口的状态随各种各样的消息而改变，所以当应用程序需要功能菜单项，决定捕捉窗口的真实状态或者调用视频格式对话框时，都应该更新这一结构中的信息。

(3) CAPDRIVERCAPS：定义了视频捕捉驱动程序的功能，如：驱动程序的数目索引是否支持视频叠加功能等。当应用程序将捕捉窗口与视频捕捉驱动程序相连接时，应该发送消息 WM_CAP_DREVER_GET_CAPS 或者调用宏 capDriverGetCaps 将驱动程序的功能拷贝一份到该结构中。

(4) VIDEOHDR：定义了视频数据块的头信息，其数据成员 lpData（指向数据缓存的指针），和 dwBufferLenth（数据缓存的大小）经常用到[5][6]。

2.4 视频处理过程设计

2.4.1 VFW软件包视频捕捉

1、创建视频捕捉窗

在进行视频捕捉之前，必须先创建一个捕获窗口[7]。在此窗口上添加操

作按纽，进行相应的操作。在窗口类里，使用函数 capCreateCaptureWindow()来创建捕获窗口。如果创建窗口成功，它将返回一个窗口句柄，否则的话，返回 NULL 值。

本程序是基于对话框的，共创建两个窗口。具体的创建窗口的程序如下：

```
CWnd *pWnd1=AfxGetMainWnd()->GetDlgItem(IDC_VIEW1);
CWnd *pWnd2=AfxGetMainWnd()->GetDlgItem(IDC_VIEW2);
hwndVideo = capCreateCaptureWindow(
                    (LPSTR) "My Capture Window",
                    WS_CHILD | WS_VISIBLE,
                    0, 0, 320,240,
                    pWnd1->GetSafeHwnd(),
                    (int) 1);
hwndAnalysis1 = capCreateCaptureWindow(
                    (LPSTR) "My Capture Window",
                    WS_CHILD | WS_VISIBLE,
                    0, 0, 640,480,
                    pWnd2->GetSafeHwnd(),
                    (int) 1);
```

其中(LPSTR)“My Capture Window”是窗口的 Caption 值；WS_CHILD | WS_VISIBLE 是捕获窗口的窗口风格，表示捕获窗口是子窗口且是可见的；"0，0"是窗口的左上角的点坐标；"320，240"设置窗口的宽度为 320，高度为 240，"640，480"设置窗口的宽度为 640，高度为 480；"pWnd1->GetSafeHwnd()"和"pWnd2->GetSafeHwnd()"是父窗口句柄，这里分别对应对话框上静态文本框 IDC_VIEW1 和 IDC_VIEW2 指定的区域；"1"是窗口的表示。

2、将捕获窗口与视频捕捉设备相联

在 AVICap 窗口类，使用 CapDriverConnect()函数来实现捕获窗口与视频设备驱动的相联。

```
if(capDriverConnect(hwndVideo,0))
{
capDriverGetCaps(hwndVideo,&m_CapDrvCap,
        sizeof(CAPDRIVERCAPS));
    if(m_CapDrvCap.fCaptureInitialized)
    {
        capGetStatus(hwndVideo,&m_CapStatus,sizeof(m_CapStatus));
        capPreviewRate( hwndVideo, 66 );
        capPreview(hwndVideo,TRUE);
     }
     else
     {
        AfxMessageBox("设备没有初始化!");
        AfxGetMainWnd()->PostMessage(WM_CLOSE);
     }
}
else
{
    AfxMessageBox("设备没连接!");
    AfxGetMainWnd()->PostMessage(WM_CLOSE);
}
```

3、视频显示

与设备关联后，需要设置视频流捕获和显示所需要的速率以及显示视频的方式。AVICap 提供了两种方式用于显示视频：预览(Preview)和叠加(Overlay)模式。本文使用的是预览模式，在预览模式中，视频帧先从捕获硬件传到系统内存，接着采用 GDI 函数在捕获窗中显示。

相关设置预览模式的程序如下：

capPreviewRate(hwndVideo,66); //set preview rate to 66 milliseconds

capPreview(hwndVideo,TRUE); //start preview video

其中 capPreviewRate 设置了预览播放速率，宏 capPreview 用于启动预览模式。

2.4.2 视频显示处理过程设计

智能车的视频显示处理过程要实现的功能有：

(1) 将实时采集到的视频图像以一定的方式存储成图片文件，用于后续的图像处理和目标识别；

(2) 将实时处理和识别的最终结果再实时地显示出来；

完成上述的两个功能要求存储方式节省磁盘空间，算法简单、可重复利用，且满足实时处理的要求。本文仅针对当前采集到的视频图像设计了一种简单的显示处理方法，其过程如下：

设置一个定时器，每隔一定时间在定时器消息响应函数中完成下述工作：

(1) 抓取视频捕捉窗口中摄像头采集到的视频，利用 AVICap 窗口类中的 CapFileSaveDIB()函数将其存成位图格式文件，并存于计算机 C 盘根目录下，命名为“Capture.bmp”。

(2) 对“C://Capture.bmp”进行一系列处理后再次保存，并将保存后的文件显示在软件界面上的视频处理窗口中。

即在定时器的每个定时间隔内，系统均要完成一次从视频图像的保存、图像处理、目标识别及生成控制算法，到再次保存处理后图像并显示在指定窗口中的过程。这样，对采集到的图像进行的一系列操作均可以实时地显示在软件界面窗口中，实现了处理结果的可视化。

2.4.3 结果及分析

按照以上的思路设计视频捕捉及显示处理过程，软件界面上视频捕捉窗口和视频处理窗口分别如图2.3和图2.4所示。

图 2.3 视频捕捉窗口

图 2.4 视频处理窗口

实际测试表明，在每一时刻，视频处理窗口中显示的图像均能快速跟踪视频捕捉窗口中视频图像的变化，且二者几乎保持同步变化，肉眼难以分辨二者的差别。由此可见，视频捕捉及处理过程的设计是较成功的，这样就为进一步的图像处理和目标识别打下了一个良好的基础。

2.5 本章小结

本章着重介绍了智能车模拟自动驾驶系统的视觉系统设计过程，包括视觉传感器的选择、视觉处理软件包VFW的应用以及视频捕捉及处理过程的算法设计，最后通过实验验证了设计的合理性。

第3章 智能车前方视野图像处理

3.1 引言

智能车辆视觉系统完成图像采集后，需要对获取的图像进行各种处理与识别。视觉系统在图像的生成、采样、量化、传输、变换等过程中，由于 CMOS 传感器的噪声、随机大气湍流、光学系统的失真等原因会造成 CMOS 摄像头成像质量的降低。另外，由于车辆行驶时视觉系统与道路环境之间存在相对运动，输出图像质量也会降低，常产生运动模糊等现象。为了改善视觉系统图像质量，需要突出道路图像中的有用信息并尽可能消除其它环境信息的干扰，因此需要对原始图像进行图像的预处理操作。

3.2 数字图像的基本概念

3.2.1 模拟图像的描述

人眼看到的任何自然界的图像都是连续的模拟图像，其形状和形态表现由图像各位置的颜色所决定。色度学理论认为，任何颜色都可由红、绿、蓝三种基本颜色按不同的比例混合得到，红、绿、蓝被称为三原色。因此，自然界的图像可用基于位置坐标的三维函数来表示[8]，即：

$$f(x,y,z)=\{f_{red}(x,y,z),f_{green}(x,y,z),f_{blue}(x,y,z)\} \tag{3.1}$$

其中 f 表示空间坐标为 (x,y,z) 位置点的颜色，f_{red}、f_{green}、f_{blue} 分别表示该点的红、绿、蓝三种原色的颜色分量值。它们都是空间的连续函数，即连续空间的每一个点都由一个精确的值与之相对应。

为了研究的方便，我们主要考虑平面图像。平面上每一点仅包括两个坐标值，因此，平面图像函数是连续的二维函数，即：

$$f(x,y)=\{f_{red}(x,y),f_{green}(x,y),f_{blue}(x,y)\} \tag{3.2}$$

图像可以分为黑白图像和彩色图像。所谓黑白图像，就是图像中的每一点都不是彩色的，即每一点的红、绿、蓝、颜色分量都相等，即：

$$f_{red} = f_{green} = f_{blue} \tag{3.3}$$

对于黑白图像，其 $f(x,y)$ 表示 (x,y) 位置处的灰度值。

3.2.2 数字图像的描述

由于计算机仅能处理离散的数据，所以如果要计算机来处理图像，连续的图像函数必须转化为离散的数据集，这一过程叫做图像采集。图像采集由图像采集系统完成，如图 3.1 所示。图像采集系统包括三个基本单元，即成像系统、采样系统和量化器。采样实际上就是一个空间坐标的量化过程，量化则是对图像函数值的离散化过程。采样和量化系统统称为数字化。

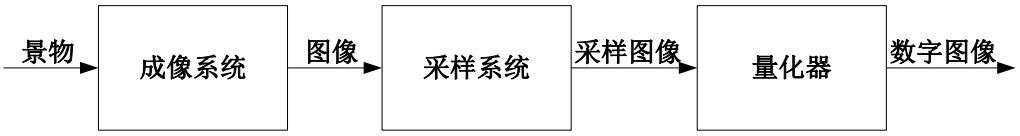

图 3.1 图像采集系统

数字图像是连续图像 $f(x,y)$ 的一种近似表示，通常由采样点的值所组成的矩阵来表示：

$$\begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,M-1) \\ f(1,0) & f(1,1) & \cdots & f(1,M-1) \\ \cdots & \cdots & \cdots & \cdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,M-1) \end{bmatrix}$$

每个采样点叫做一个像素(pixel)。上式中，M ，N 分别为数字图像在横、纵方向上的像素数。在计算机中通常采用二维数组来表示数字图像的矩阵。

把像素按不同的方式进行组织或存储，就得到不同的图像格式，把图像数据存成文件就得到图像文件。在 Windows 系统中，最常用的图像格式是位图格式，其文件名以 BMP 为扩展名。

图像数字化的精度包括两个部分，即分辨率和颜色深度。分辨率是指图像数字化的空间精细度，有显示分辨率和图像分辨率两种不同的分辨率。图像分辨率实质是数字化图像时划分的图像的像素密度，即单位长度内的像素数，其单位是每英寸的点数 DPI(Dots per Inch)。显示分辨率则是把数字图像

在输出设备（如显示器或打印机）上能够显示的像素数目和所显示像素之间的点距。图像分辨率说明了数字图像的实际精细度，显示分辨率说明了数字图像的表现精细度。具有不同的图像分辨率的数字图像在同一设备上的显示分辨率相同。显示器是常见的图像输出设备，现在常见的显示器的分辨率一般可达 1024×768，点距为 0.28mm。

3.2.3 数字图像的颜色表示方法

数字图像的颜色深度是指表示每一像素的颜色值的二进制位数。颜色深度越大则能表现的像素的颜色数目越多，它们之间的关系取决于数字图像采用的颜色表示法。常用的颜色表示法有 RGB、CMYK、HSL 和 YUV 等。

用 R、G、B 颜色分量来表示数字图像像素的颜色值的方法就是 RGB 法。如果表示 R、G、B 颜色分量的位数分别为n_1、n_2、n_3，则可表示的像素的颜色数是$2^{(n_1+n_2+n_3)}$。如分别用 8 位来表示三颜色分量，则总共需要 24 位来表示 RGB 三色，可表示的颜色数为$2^{(8+8+8)}=2^{24}=16{,}777{,}216$。

CMYK（Cyan Magenta Yellow Black, 青、紫红、黄、黑）法多用于印刷。物理上，青光吸收红光反射绿光和蓝光；紫红光吸收绿光，反射蓝光和红光；黄光吸收蓝光，反射红光和绿光。数字图像文件在内存中的存储方式是 RGB 的值，但由 RGB 值转化为 CMY 值时，颜色并与原来相同。因此，为了产生正确的 CMY 的值，必须找到隐含在 RGB 值中的灰度，并转为黑色，所以 K 值是必不可少的。

物体的颜色还可通过色调(Hue)、饱和度(Saturation)和亮度(Luminance)的不同而表现出来，这种表示法叫做 HSL 表示法。色调表示基本的纯色，饱和度的数值表示颜色中掺入白光的比例，亮度则表示颜色中掺入黑光的比例。此方法适合于人的直觉的配色方案，只需要选择色调、色度、亮度，就可以方便的配出所需要的颜色。

YUV 表示法则是另一种常用的颜色表示法，主要用于多媒体计算机技术中。其基本特征是将亮度信号与色度信号分离表示，Y 代表亮度，U、V 是

两个彩色分量，表示色差，一般是蓝、红的相对值。由于人眼对亮度的变化比对颜色的变化更敏感，因此 YUV 模型中 Y 分量的值所占带宽大于或等于彩色分量所占的带宽。主要的格式有 4:1:1、4:2:2、4:4:4 等三种。YUV 和 RGB 模型之间的线性关系如下：

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.301 & 0.586 & 0.113 \\ -0.301 & -0.586 & 0.887 \\ 0.699 & -0.586 & -0.113 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.4}$$

实际上，U 、V 的值是由 B 、 R 和 Y 来确定的，即

$$\begin{aligned} U &= B - Y \\ V &= R - Y \end{aligned} \tag{3.5}$$

CMYK、HSL 与 RGB 表示法也可以按一定的算法进行转换，其中 RGB 是数字图像处理的主要表示法。

本论文所研究的地图图像都是 Windows 系统中的 BMP 位图，以 RGB 颜色表示法表示。

3.3 图像的灰度化处理

3.3.1 灰度化原理

颜色可分为黑白和彩色。黑白颜色是指颜色中不包含任何的彩色的成分，仅由黑色和白色组成。在 RGB 颜色模型中，如果 R=G=B，则颜色(R，G，B)表示一种黑白颜色，其中 R=G=B 的值叫做灰度值。由彩色转化为灰度的过程称为灰度化处理。灰度化就是使彩色的 R、G、B 分量相等的过程。由于 R、G、B 的取值范围是 0~255，所以灰度的级别只有 256 级，即灰度图像只能表现 256 种颜色。

灰度化处理的方法主要有如下三种：

(1) 最大值法：使 R、G、B 的值等于三个值中最大的一个，即：

$$R = G = B = \max(R, G, B) \tag{3.6}$$

最大值法会形成亮度很高的灰度图像。

(2) 平均值法：对 R、G、B 求出平均值，即：

$$R = G = B = (R, G, B)/3 \tag{3.7}$$

平均值法会形成比较柔和的灰度图像。

(3) 加权平均值法：根据重要性或其他指标给 R、G、B 赋予不同的权值，并使 R、G、B 它们的值加权，即：

$$R = G = B = (W_R R + W_G G + W_B B)/(W_R + W_G + W_B) \tag{3.8}$$

其中，W_R、W_G、W_B 分别为 R、G、B 的权值。W_R、W_G、W_B 取不同的值，加权平均值法将形成不同的灰度图像。由于人眼对绿色的敏感度最高，对红色的敏感度次之，对蓝色的敏感度最低，因此使$W_G > W_R > W_B$将得到比较合理的灰度图像。实验和理论推导证明，当W_R=0.30，W_G=0.59，W_B=0.11 时，即：

$$R = G = B = 0.3R + 0.59G + 0.11B \tag{3.9}$$

此时，R、G、B 的取值就是该像素的亮度值，此时，得到的灰度图像最合理。

3.3.2 处理及结果分析

本设计中摄像头采集到的图像是 24 位真彩色图像，本文采用第三种方法——加权平均法来对采集到的原始图像进行灰度化处理。首先利用公式(3.9)将 24 位真彩色图像转化成 24 位灰度图像，然后再将其压缩为 8 位灰度图像。灰度化处理的结果如图 3.2 所示。

图 3.2 灰度化处理结果图

3.4 图像的平滑滤波处理

3.4.1 线性平滑滤波处理——均值滤波

图像平滑滤波处理分为线性滤波与非线性滤波，线性滤波方法提出较早，且具有较完备的理论基础。针对线性滤波处理，本文主要尝试了均值滤波中值滤波的方法。

均值滤波是对图像进行局部均值运算，每一个像素值用其局部邻域内所有值的均值置换，即：

$$g(i,j)=\frac{1}{(2K+1)(2L+1)}\sum_{m-i-K}^{i+K}\sum_{n-j-L}^{j+L}f(m,n) \tag{3.10}$$

其中 $f(m,n)$ 为一幅 $M\times N$ 的道路图像，均值滤波窗口为 $(2K+1)(2L+1)$，窗口在两个方向上都必须为奇数，否则图像会产生偏移。均值滤波器可以通过卷积模板的等权值卷积运算来实现，本文采用了 $K=1,L=1$ 的八邻域均值滤波模板来完成均值滤波处理。均值滤波可以去除图像噪声等高频成分，但是同时也会导致图像细节的损失，图像的模糊程度会更严重。为了克服这一缺点，采用阈值法减少由于邻域平均所产生的模糊效应，其基本方法由式(3.11)决定：

当 $\left|f(i,j)-\frac{1}{(2K+1)(2L+1)}\sum_{m-i-j}^{i+K}\sum_{n-j-L}^{j+L}f(m,n)\right|>T$ 时

$$g(i,j)=\frac{1}{(2K+1)(2L+1)}\sum_{m-i-K}^{i+K}\sum_{n-j-L}^{j+L}f(m,n)$$

否则，

$$g(i,j)=f(m,n) \tag{3.11}$$

其中 T 为规定的非负阈值，阈值法均值滤波处理的途径是当一些点和它的邻域内的点的灰度平均值的差不超过规定的阈值 T 时，就仍然保留其原灰度值不变，如果大于阈值 T 时就用它们的平均值来代替该点的灰度值，这样就可以大大减少图像模糊程度。

3.4.2 非线性平滑滤波处理——中值滤波

理论和实验证明，虽然线性滤波具有良好的噪声抑制能力，但是对图像的平滑会造成的图像中的细节信息损失，从而使处理后的图像产生模糊。

中值滤波是一种非线性的图像滤波方法，它于 1971 年由 J.W.Jukey 提出，并首先应用于一维信号处理技术中，后来被二维图像处理技术所采用。中值滤波的基本思想是用像素点邻域灰度值的中值来代替该像素点的灰度值（如图 3.3 所示），该方法在去脉冲噪声、椒盐噪声的同时又能保留图像边缘细节。

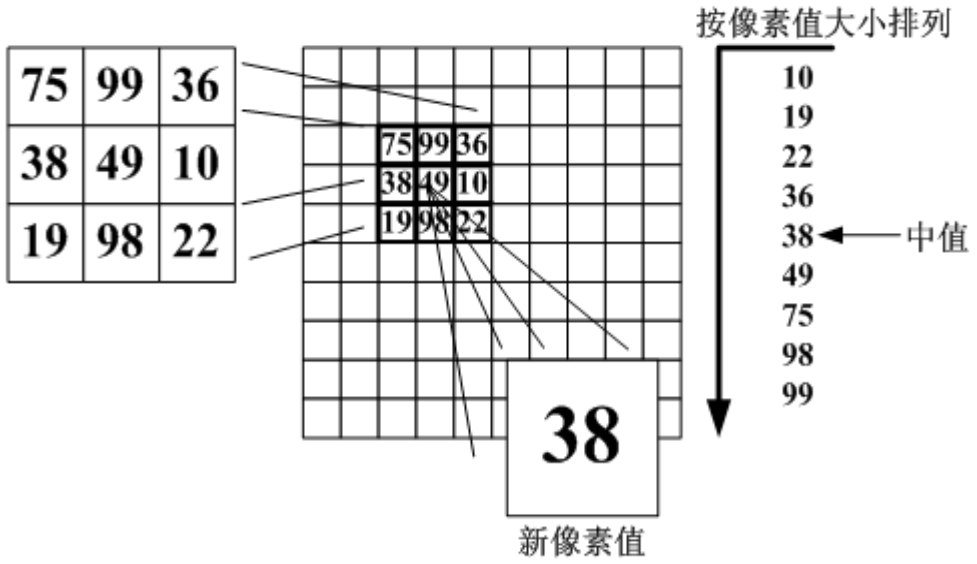

图 3.3 中值滤波的基本思想

中值滤波不影响阶跃函数和斜坡函数，并且可以有效消除单、双脉冲、使三角函数的顶端变平（见图 3.4 所示）。中值滤波在一定条件下可以克服线性滤波所带来的图像细节模糊，同时在实际运算过程中并不需要图像的统计特征，也给计算带来不少方便，所以本文尝试了中值滤波平滑处理方法。

具体做法是：首先确定一个含有奇数个像素 $(2n+1)\times(2n+1)$ 的窗口 W，图像大小为 $M\times N$（见图 3.3 所示），窗口内各像素按照灰度大小排队后，用其中间位置的灰度值代替原 $f(x,y)$，得到增强后的图像 $g(x,y)$，可以表示为：

$$g(x,y)=median\{f(x-k,y-l),\quad (k,l)\in W\} \tag{3.12}$$

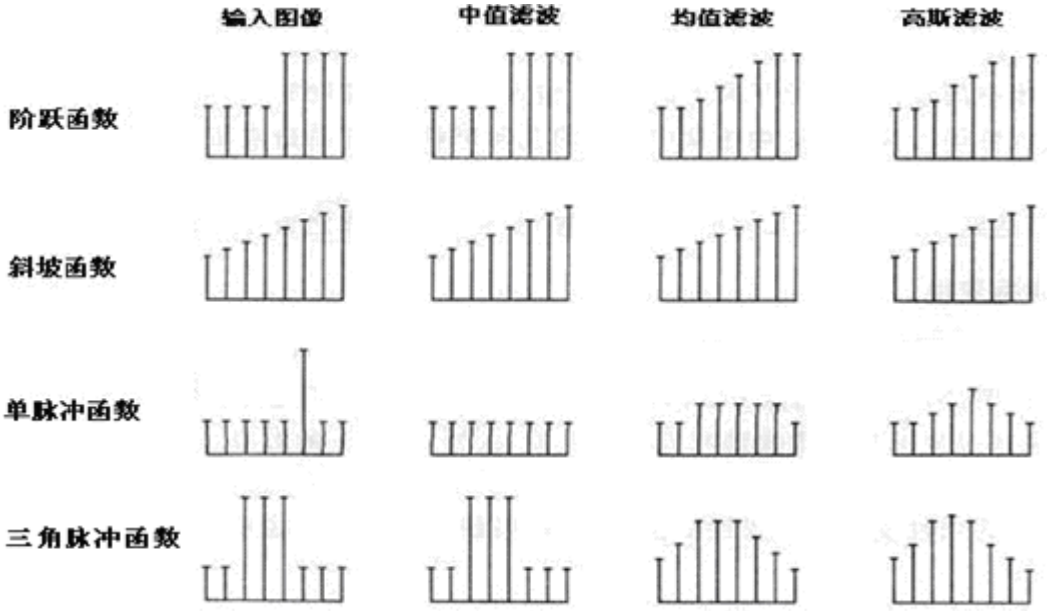

图 3.4 不同滤波方法处理结果比较

3.4.3 处理及结果分析

本文选择的窗口W大小为 3×3，道路图像中值滤波处理的试验结果如图 3.5 所示。

从处理结果可以看出中值滤波平滑化了图像，削弱了一部分锐化的噪声。

图 3.5 中值滤波处理结果图

3.5 图像的边缘检测

利用计算机进行图像处理有两个目的：一是产生更适合人观察和识别的

图像；二是希望能由计算机自动识别和理解图像。

无论为了哪种目的，图像处理中关键的一步就是对包含有大量各式各样景物信息的图像进行分解。分解的最终目的是图像被分解成一些具有某种特征的最小成分，称为图像的基元。相对于整幅图像来说，这种基元更容易被快速处理。

图像的特征指图像场中可用作标志的属性。它可以分为图像的统计特征和图像的视觉特征两类。

图像的统计特征是指一些人为定义的特征，通过变换才能看到，如图像的直方图、矩、频谱等等；图像的视觉特征是指人的视觉可直接感受到自然特征，如区域的亮度、纹理或轮廓等。利用这两类特征把图像分解成一系列有意义的目标或区域的过程称为图像的分割。

图像的边缘是图像的最基本特征。

所谓边缘（或边沿）是指其周围像素灰度有阶跃变化或屋顶变化的那些像素的集合。边缘广泛存在于物体与背景之间、物体与物体之间、基元与基元之间。因此，它是图像分割所依赖的重要特征[9]。

在本节中，本文将介绍图像边缘的检测和提取技术。

3.5.1 边缘检测的原理

物体的边缘（或边沿）是由灰度不连续性所反映的。经典的边缘提取方法是考察图像的每个像素在某个邻域内灰度的变化，利用边缘临近一阶或二阶方向导数变化规律，用简单的方法检测边缘。这种方法称为边缘检测局部算子法。

边缘的种类可以分为两种：一种称为阶跃性边缘，它两边的像素的灰度值有着显著的不同；另一种成为屋顶状边缘，它位于灰度值从增加到减少的变化转折点。

图 3.6 中分别给出了这两种边缘的示意图及相应的一阶方向导数、 二阶方向导数的变化规律。对于阶跃性边缘，二阶方向导数在边缘处呈零交叉；

而对于屋顶状边缘，二阶方向导数在边缘处取极值[10]。

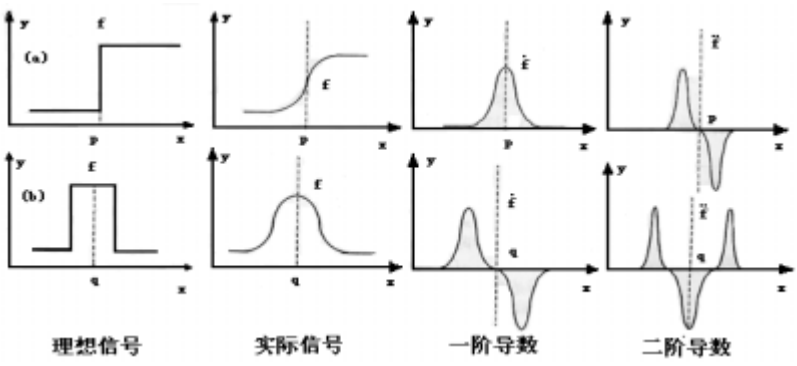

图 3.6 两种边缘示意图及相应方向导数

从“边缘是亮度急剧变化的地方”这一出发点考虑，为了提取边缘要素最直接的方法是利用亮度变化的微分值[11]。这里所说的微分是由 x 和 y 坐标对图像的灰度变化进行微分的空间微分。现在，设 $x-y$ 坐标上的图像的亮度为 $f(x,y)$，则空间一次微分(gradient)可以用式(3.13)的矢量来表达：

$$\nabla f(x,y)=\frac{\partial f(x,y)}{\partial x}u_x+\frac{\partial f(x,y)}{\partial y}u_y \tag{3.13}$$

其中，u_x、u_y 分别是 x、y 方向上的单位矢量。

将 x 方向的微分用 f_x，y 方向的微分用 f_y 表示，则一次微分的强度值（一次微分值）为

$$\left|\nabla f(x,y)\right|=\sqrt{{f_x}^2(x,y)+{f_y}^2(x,y)} \tag{3.14}$$

可以认为，这是相当于边缘强度的量。另一方面，二次微分(laplacian)由式(3.15)给出：

$$\nabla^2 f(x,y)=\frac{\partial^2 f(x,y)}{\partial_x{}^2}+\frac{\partial^2 f(x,y)}{\partial_y{}^2}=f_{xx}(x,y)+f_{yy}(x,y) \tag{3.15}$$

其中，$f_{xx}(x,y)$ 和 $f_{yy}(x,y)$ 分别是 x 方向和 y 方向的二次微分。

在现实的数字图像中，x、y 坐标的值只能是整数，这样上面的一般数学表达式就不适用了。在很多时候，不采用微分而是采用差分。使用相邻像素间的灰度级差是最简单的方法。即如果设 $f(i,j)$ 为在坐标 (i,j) 位置的灰度

级，则一次微分可以用式(3.16)计算：

$$\begin{aligned} f_x(i,j) &= f(i+1,j) - f(i,j) \\ f_y(i,j) &= f(i,j+1) - f(i,j) \end{aligned} \tag{3.16}$$

这样得到的 $f_x(i,j)$、$f_y(i,j)$ 严格来说不是坐标 (i,j) 的一次微分，而是偏了半个像素的坐标 $(i+0.5, j+0.5)$，即像素和像素的分界地方的一次微分。这时候，如果计算机内的数据排列和空间坐标发生偏差的时候就很难使用了。为此，一般隔一个像素进行差分运算，即

$$\begin{aligned} f_x(i,j) &= f(i+1,j) - f(i-1,j) \\ f_y(i,j) &= f(i,j+1) - f(i,j-1) \end{aligned} \tag{3.17}$$

这些差分可以通过权重系数矩阵与图像数据之间的积和运算来算出。前式的 f_x 中，可以进行.1 0 1 与数字图像中的所有 x 方向的恶 3 个像素的灰度级之间的积和运算，即

$$f_x(x,y) = (-1)\times f(i-1,j) + 0\times f(i,j) + (+1)\times f(i+1,j)$$

将此记述为

$$f_x：\ [-1 \quad 0 \quad 1]$$

y 方向的一次微分为

$$f_y(x,y) = (-1)\times f(i,j-1) + 0\times f(i,j) + (+1)\times f(i,j+1)$$

可以表示为

$$f_y：\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

分别扩大到3×3的区域，则为如下形式：

$$f_x：\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix},\quad f_y：\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

这样就可以得到在一次微分中应用最为频繁的局部积和运算权重系数矩阵。也就是

$$\begin{aligned} f_x(i,j) &= [f(i+1,j+1)+f(i+1,j)+f(i+1,j-1)] \\ &\quad -[f(i-1,j+1)+f(i-1,j)+f(i-1,j-1)] \\ f_y(i,j) &= [f(i-1,j-1)+f(i,j-1)+f(i+1,j-1)] \\ &\quad -[f(i-1,j+1)+f(i,j+1)+f(i+1,j+1)] \end{aligned} \tag{3.18}$$

二次微分也可以通过差分来计算。这时，可以利用像素和像素分界的差分值来求出：

$$\begin{aligned} f_{xx}(i,j) &= [f(i+1,j)-f(i,j)]-[f(i,j)-f(i-1,j)] \\ &= f(i+1,j)-2\times f(i,j)+f(i-1,j) \\ f_{yy}(i,j) &= [f(i,j+1)-f(i,j)]-[f(i,j)-f(i,j-1)] \\ &= f(i,j+1)-2\times f(i,j)+f(i,j-1) \end{aligned} \tag{3.19}$$

系数是

$$f_{xx}=[1 \quad -2 \quad 1],\quad f_{yy}=\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

将这些进行线性结合就可以导出拉普拉斯滤波的系数：

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

将45°角的方向也考虑进去的下式是最常用的：

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

3.5.2 边缘检测的算法

符号及说明：

M --待检测图像像素矩阵

$f(x,y)$--待检测图像(x,y)坐标点处的像素值

G_X--总边缘检测运算结果（Roberts 算子 X 用 R 替换；Sobel 算子 X 用 S 替换；Prewitt 算子 X 用 P 替换；Krisch 算子 X 用 K 替换；Laplacian 算子 X 用

L 替换）

X_x -- x 方向边缘检测卷积算子（Roberts 算子 X 用 R 替换；Sobel 算子 X 用 S 替换；Prewitt 算子 X 用 P 替换；Krisch 算子 X 用 K 替换；Laplacian 算子 X 用 L 替换）

X_y -- y 方向边缘检测卷积算子（Roberts 算子 X 用 R 替换；Sobel 算子 X 用 S 替换；Prewitt 算子 X 用 P 替换；Krisch 算子 X 用 K 替换；Laplacian 算子 X 用 L 替换）

1、Roberts 边缘算子

Roberts 边缘算子是一种利用局部差分算子寻找边缘的算子。如果本文用 Roberts 算子检测图像 M 的边缘的话，本文可以先分别用水平算子和垂直算子对图像进行卷积，得到的是两个矩阵，在不考虑边界的情形下也是和原图像同样大小的 G_{Rx}，G_{Ry}，它们分别表示图像 M 中相同位置处的两个偏导数。然后把 G_{Rx}，G_{Ry} 对应位置的两个数平方后相加再开方得到一个新的矩阵 G，G 表示 M 中各个像素的灰度的梯度值的逼近。它们分别由式(3.20)和式(3.21)给出：

$$\begin{cases} G_{Rx} = R_x \otimes M = f(x,y) - f(x+1,y+1) \\ G_{Ry} = R_y \otimes M = f(x,y+1) - f(x+1,y) \end{cases} \tag{3.20}$$

$$G_R = \sqrt{G_{Rx}{}^2 + G_{Ry}{}^2} \tag{3.21}$$

其中 $f(x,y)$ 是具有整数像素坐标的输入图像，平方根的运算使该处理类似在人类视觉系统中发生的过程。Roberts 算子 R_x 与 R_y 可以用模板表示，如图 3.7 所示：

1	**0**
0	**-1**

（1）R_x

0	**-1**
1	**0**

（2）R_y

图 3.7 Roberts 边缘算子

2、Sobel 边缘算子

Sobel 边缘算子是一阶微分算子，它是一种将方向差分运算与局部平均相结合的方法。该算子在以 $f(x,y)$ 为中心的 3×3 邻域上计算 x 和 y 方向上的偏导数，即：

$$\begin{cases} G_{Sx}=\left[f(x+1,y-1)+2f(x+1,y)+f(x+1,y+1)\right]- \\ \qquad \left[f(x-1,y-1)+2f(x-1,y)+f(x-1,y+1)\right] \\ G_{Sy}=\left[f(x-1,y+1)+2f(x,y+1)+f(x+1,y+1)\right]- \\ \qquad \left[f(x-1,y-1)+2f(x,y-1)+f(x+1,y-1)\right] \end{cases} \tag{3.22}$$

此时 S_x 和 S_y 可用卷积模板来表示，如图 3.8 所示：

-1	**-2**	**-1**
0	**0**	**0**
1	**2**	**1**

（1）S_x

-1	**0**	**1**
-2	**0**	**2**
-1	**0**	**1**

（2）S_y

图 3.8 Sobel 边缘算子

3、Prewitt 边缘算子

Prewitt 边缘算子也是一阶微分算子，与 Sobel 算子类似，也是采用先求平均再求差分的方法，因而可以抑制噪声，增强图像的边缘特征。Prewitt 算子法也是采用类似的计算偏微分估计值的方法：

$$\begin{cases} P_x=\left[f(x+1,y-1)+f(x+1,y)+f(x+1,y+1)\right]- \\ \qquad \left[f(x-1,y-1)+f(x-1,y)+f(x-1,y+1)\right] \\ P_y=\left[f(x-1,y+1)+f(x,y+1)+f(x+1,y+1)\right]- \\ \qquad \left[f(x-1,y-1)+f(x,y-1)+f(x+1,y-1)\right] \end{cases} \tag{3.23}$$

此时 P_x 和 P_y 可用卷积模板来表示，如图 3.9 所示：

-1	-1	-1
0	0	0
1	1	1

（1）P_x

-1	0	1
-1	0	1
-1	0	1

（2）P_y

图 3.9 Prewitt 边缘算子

4、Krisch 边缘算子

如图 3.10 所示的 8 个卷积核组成了 Krisch 边缘算子。图像中的每个点都用 8 个掩模进行卷积，每个掩模都对某个特定边缘方向作出最大响应，所有 8 个方向中的最大值作为边缘幅度图像的输出。最大响应掩模的序号构成了边缘方向的编码。

+5	+5	+5
-3	0	-3
-3	-3	-3

-3	+5	+5
-3	0	+5
-3	-3	-3

-3	-3	+5
-3	0	+5
-3	-3	+5

-3	-3	-3
-3	0	+5
-3	+5	+5

-3	-3	-3
-3	0	-3
+5	+5	+5

-3	-3	-3
+5	0	-3
+5	+5	-3

+5	-3	-3
+5	0	-3
+5	-3	-3

+5	+5	-3
+5	0	-3
-3	-3	-3

图 3.10 Krisch 边缘检测算子

5、Laplacian 边缘算子

Laplacian 边缘算子是对二维函数进行运算的二阶导数算子。通常使用的 Laplacian 算子如图 3.11 所示：

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

图 3.11 Laplacian 边缘检测算子

3.5.3 处理及结果分析

本文采用 Sobel 边缘算法来提取道路边缘， Sobel 边缘提取的结果如图 3.12 所示。

从处理结果可以看出图像中所有的边缘基本都被提取出来了，如图 3.12 中白线所示，其中包括道路边缘。

图 3.12 Sobel 边缘提取处理结果图

3.6 图像的阈值分割处理

3.6.1 最大最小方差阈值分割法

道路图像经过平滑滤波、边缘增强处理后，仍然包含有大量的环境干扰信息。为了减少这些无用信息的干扰，减少道路图像识别的数据处理量，突出道路图像中有用的边缘特征，提高识别的可靠性和实时性，还需要进行阈值分割处理，即二值化处理。

道路图像的阈值分割处理中，阈值选择是一个关键问题。阈值选取的合理与否直接影响道路识别的结果。理想的阈值应该是最大程度的突出道路图像中有用的边界特征而抑制环境干扰。设道路图像的灰度级为$G=\{0,1,2,\cdots\}$，阈值$T\in G$，二值灰度集合$B=\{b_0,b_1\}$，则道路图像的阈值分割处理可表示为：

$$f_T(x,y)=\begin{cases}b_0 & 若f(x,y)>T\\ b_1 & 若f(x,y)\le T\end{cases} \tag{3.24}$$

选取不同的阈值T将产生不同的二值图像，为了获得较好的图像分割效果，常需要基于某种准则选择最佳的阈值T。选取阈值的准则分为基于图像

像素邻域局部特性的局部阈值法和用同一个阈值来分割整幅图像的全局阈值法。为了较好的适应外界环境的变化，智能车辆道路图像采用全局自动阈值分割法。

全局阈值自动选取方法很多，有最佳阈值分割简单迭代法、最小错误率分割法与最大最小方差法等，针对本文需要处理的道路图像，考虑到处理的实时性和有效性要求采用了最大最小方差法的阈值分割方法。

这种阈值分割方法是根据模式识别中的分类思想得到的，由日本的大津展之提出，也称为大津法。对道路图像进行阈值分割，就是把图像直方图从某一阈值处分割为两类：目标区域和背景区域，而这种方法的基本思想是使这两类的类内方差最小，而两类间的方差最大。

假设道路图像的直方图灰度范围在$[m_1,m_2]$之间，用一整数$K\in(m_1,m_2)$将直方图分为C_1和C_2两类，其直方图范围分别为$[m_1,K]$和$[K,m_2]$。图像的总像素数为N，灰度值为i的像素数为n_i，各像素值出现的概率为P_i，并用ω_1,ω_2表示C_1,C_2类产生的概率，μ_1,μ_2表示C_1,C_2类的均值，则

$$N=\sum_{i=m_1}^{m_2}n_i \qquad P_i=\frac{n_i}{N} \tag{3.25}$$

$$\omega_1=\sum_{i=m_1}^{K}P_i \qquad \mu_1=\frac{\sum_{i=m_1}^{K}i\cdot P_i}{\omega_1}$$
$$\omega_2=\sum_{i=K+1}^{m_2}P_i \qquad \mu_2=\frac{\sum_{i=K+1}^{m_2}i\cdot P_i}{\omega_2} \tag{3.26}$$

根据两类的产生概率、均值与整幅道路图像的灰度统计值μ的关系有$\mu=\omega_1\mu_1+\omega_2\mu_2$，其中显然有$\omega_1+\omega_2=1$，设$N_1,N_2$分别表示$C_1,C_2$类的像素数，两类间的方差${\sigma_{1,2}}^2$及两类内的方差${\sigma_1}^2,{\sigma_2}^2$为

$$\begin{aligned}{\sigma_{1,2}}^2&=\omega_1(\mu_1-\mu)^2+\omega_2(\mu_2-\mu)^2\\&=\omega_1\omega_2(\mu_1-\mu_2)^2\end{aligned} \tag{3.27}$$

$$\sigma_1^{\,2} = \sum_{i=m_1}^{K} P_1(i)(i-\mu_1)^2$$
$$\sigma_2^{\,2} = \sum_{i=K+1_1}^{m_2} P_2(i)(i-\mu_2)^2 \tag{3.28}$$

其中 $P_j(i) = n_i / N_j$，表示 C_j 类灰度值为 i 的像素出现的概率，$j = 1,2$。由式(3.27)与(3.28)可知类内方差与类间方差均为 K 的函数，当选取 $[m_1, m_2]$ 间的某值 K，使得两类间方差取最大值而两类内方差取最小值，此时的 K 即为最佳阈值。为了满足这一要求，构造一个判别函数 J：

$$J = \frac{\sigma_{1,2}}{\sigma_1 + \sigma_2} \tag{3.29}$$

由上式可知，使判别函数 J 取最大值的 K 值即为所求的阈值。这种阈值分割方法不涉及先验概率的问题，所以这种阈值分割法能够将道路图像中的背景区域与目标区域尽可能的分开。

3.6.2 处理及结果分析

道路图像经过最大最小方差阈值分割法处理的结果如图 3.13 所示。

从处理结果可以看出，阈值分割处理突出了图像中所有的边缘特征，包括道路边缘。图像经阈值分割后，所有边缘部分基本上均为白色，其余部分为黑色。道路边缘内部的白点为噪声点，是因为光照和道路污染等原因被误认为边缘，在后续处理和识别中应设法排除或弱化这一干扰。

图 3.13 最大最小方差阈值分割法处理结果图

为了便于进一步的处理，对图 3.13 的图像做反色变换，结果如图 3.14 所示：

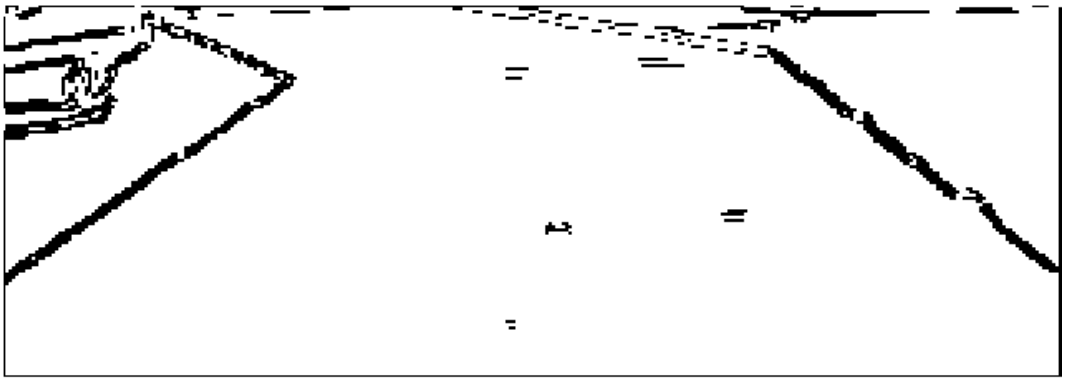

图 3.14 反色变换处理结果图

3.7 数学形态学理论

数学形态学(Mathematical Morphology)是一种应用于图像处理和模式识别领域的新的方法。形态学是生物学的一个分支，常用来处理动物和植物的形状和结构。数学形态学是建立在严格的数学理论基础上的科学。用于描述数学形态学的语言是集合论，利用数学形态学对物体几何结构的分析过程就是主客体相互逼近的过程。利用数学形态学的几个基本概念和运算，将结构元素灵活的组合、分解，应用形态变换序列达到分析的目的。

3.7.1 数学形态学的基本概念和定义

1、数学形态学的基本概念

在数学意义上，本文用形态学来处理一些图像，用以描述某些区域的形状如边界曲线、骨架结构和凸形外壳。另外，本文也可用形态学技术进行预测和快速处理如形态过滤，形态细化，形态修饰等。而这些处理都是基于一些基本运算实现的。

用于描述形态学的语言是集合论。集合代表图像中物体的形状，例如：在二值图像中所有的黑色像素点的集合就是这幅图像的完整描述。在二值图像中，当前集合是指二维整形空间的成员，集合中的每个元素就是一个二维

变量，用(x,y)表示。按规则代表图像中的一个黑色像素点。灰度数字图像可以用三维集合来表示。在这种情况下，集合中每个元素的前两个元素表示像素点的坐标，第三个变量代表离散的灰度值。在更高维的空间集合中可以包括其他的图像属性，如颜色和时间。

形态学运算的质量取决于所选取的结构元素和形态变换。结构元的选择要根据具体情况来确定，而形态运算的选择必须满足一些基本的约束条件。这些约束条件称为图像定量分析的原则。下面列出了数学形态学的几条定量分析原则：

(1) 平移不变性

设待分析的图像为X，Φ表示某种图像变换或运算，$\Phi(X)$表示X经变换或运算后的新图像。设h为一矢量，X_h表示将图像X平移一个位移矢量后的结果，那末，平移不变性原则可表示为：

$$\Phi(X_h)=\left[\Phi(X)\right]_h \tag{3.30}$$

此式说明，图像X先平移然后变换的结果与图像先变换后平移的结果相一致。

(2) 尺度变换不变性

设缩放因子λ是一个正的实常数，λX表示对图像X所做的相似变换，则尺度变换不变性可表示如下：

$$\lambda\Phi(\frac{1}{\lambda}X)=\Phi_\lambda(X) \tag{3.31}$$

如果设图像运算Φ为结构元素B对X的腐蚀（记为$X\Theta B$），则Φ_λ为结构元素λB对X的腐蚀，则上式具体化为

$$\lambda\Phi(\frac{1}{\lambda}X\Theta B)=X\Theta\lambda B \tag{3.32}$$

(3) 局部知识原理

如果Z是一个图形（闭集），则相对于Z存在另一个闭集Z'，使得对于图形X有下式成立：

$$(\Phi(X\bigcap Z))\bigcap Z'=\Phi(X)\bigcap Z' \tag{3.33}$$

可以将 Z 理解为一个“掩模”。在实际中，观察某一个对象时，每次只能观察一个局部，即某一掩模覆盖的部分。该原则要求对每种确定的变换或运算 Φ，当掩模 Z 选定以后，都能找到一个相应的模板 Z',使得通过 Z' 所观察到的局部性质，即 $(\Phi(X \bigcap Z)) \bigcap Z'$ 与整体性质 $\Phi(X) \bigcap Z'$ 相一致。

(4) 半连续原理

在研究一幅图像时，常采用逐步逼近的方法，即对图像 X 的研究往往需要通过一系列图像 $X_1, X_2, \cdots, X_n, \cdots$的研究实现，其中诸个 X_n 逐步逼近 X 。半连续原理要求各种图像变换后应满足这样的性质：对真实图像 X 的处理结果应包含在对一系列图像 X_n 的处理结果内。

(5) 形态运算的基本性质

除了一些特殊情况外，数学形态学处理一般都是不可逆的。实际上，对图像进行重构的思想在该情况下是不恰当的。任何形态处理的目的都是通过变换法去除不感兴趣的信息，保留感兴趣的信息。在形态运算中的几个关键性质如下：

递增性： $X \subset Y \Rightarrow \Phi(X) \subset \Phi(Y) \quad \forall X, Y \in \xi(E)$ (3.34)

反扩展性： $\Phi(X) \subset X \quad \forall X \in \xi(E)$ (3.35)

幂等性： $\Phi[\Phi(X)] = \Phi(X)$ (3.36)

其中： $\xi(E)$ 表示欧几里得(Euclidean)空间 E 的幂级。

2、数学形态学的基本定义

集合论是数学形态学的基础，在这里本文首先对集合论的一些基本概念作一总结性的概括介绍。对于形态处理的讨论，本文将从两个最基本的模加处理和模减处理开始。它们是以后大多数形态处理的基础。

(1) 集合

具有某种性质的确定的有区别的事物的全体。如果某种事物不存在，称为空集。集合常用大写字母 $A, B, C, \cdots$表示，空集用 ϕ 表示。

设 E 为一自由空间， $\xi(E)$ 是由集合空间 E 所构成的幂集，集合 $X, B \in \xi(E)$，则集合 X 和 B 之间只能有以下三种形式（如图 3.15 所示）：

①集合 B 包含于 X （表示为 $B \subset X$ ）；

②集合 B 击中 X （表示为 $B \Uparrow X$ ），即 $B \bigcap X \neq \phi$ ；

③集合 B 相离于 X （表示为 $B \subset X^c$ ），即 $B \bigcap X = \phi$ ；

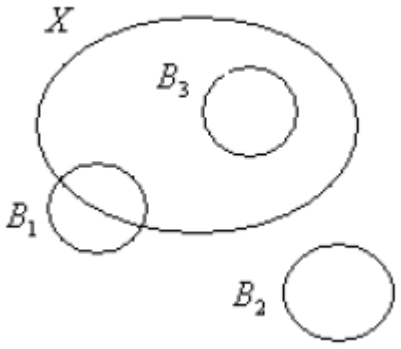

图 3.15 B_1 击中 X ， B_2 相离于 X ， B_3 包含于 X

(2) 元素

构成集合的每一个事物称之为元素。元素常用小写字母 a ，b ，c …表示，应注意的是，任何事物都不是空集的元素。

(3) 平移转换

设 A 和 B 是两个二维集合， A 和 B 中的元素分别是

$$a = (a_1, a_2) \quad b = (b_1, b_2) \tag{3.37}$$

定义 $x = (x_1, x_2)$ ，对集合 A 的平移转换为

$$(A)_x = \{c \mid c = a + x, a \in A\} \tag{3.38}$$

(4) 子集

当且仅当集合 A 的所有元素都属于 B 时，称 A 是 B 的子集。

(5) 补集

定义集合 A 的补集为：

$$A^c = \{x \mid x \notin A\} \tag{3.39}$$

(6) 差集

定义集合 A 和 B 的差集为：

$$A - B = \{x \mid x \in A, x \notin B\} = A \bigcap B^c \tag{3.40}$$

（7）映像

定义集合 A 的映像为 $\hat{A}$，定义为：

$$\hat{A}=\{x \mid x=-a, a \in A\} \tag{3.41}$$

（8）并集

由 A 和 B 的所有元素组成的集合称为 A 和 B 的并集。

（9）交集

由 A 和 B 的公共元素组成的集合称为 A 和 B 的交集。

3.7.2 二值形态学的基本运算

1、**腐蚀**(Erosion)

集合 A 被集合 B 腐蚀，表示为 $A \Theta B$，其定义为

$$A \Theta B=\{x, B+x \subset A\} \tag{3.42}$$

其中 A 为输入图像，B 称为结构元素。$A \Theta B$ 由将 B 平移 x 但仍包含在 A 内的所有点组成。如果将 B 看做为模板，那么，$A \Theta B$ 则由在平移模板的过程中，所有可以填入 A 内部的模板的原点组成。如果模板原点在结构元素的内部，那么，腐蚀具有收缩输入图像的作用。

腐蚀除了可以用式(3.42)填充形式的方程定义外，还有一个更重要的表达式：

$$A \Theta B=\bigcap\{A-b: b \in B\} \tag{3.43}$$

这里腐蚀可以通过将输入的图像平移 $-b$（b 属于结构元素），并计算所有平移的交集而得到。

2、**膨胀**(Dilation)

膨胀是腐蚀运算的对偶运算（逆运算），可以通过对补集的腐蚀来定义。集合 A 被集合 B 膨胀表示为 $A \oplus B$，其定义为：

$$A \oplus B=\left[A^{c} \Theta(-B)\right]^{c} \tag{3.44}$$

其中，A^{c} 表示 A 的补集。为了利用 B 膨胀 A，可将 B 相对于原点旋转 180° 得到 $-B$，再利用 $-B$ 对 A^{c} 进行腐蚀。腐蚀结果的补集便是所求的结果。

因为膨胀是利用结构元素对图像补集进行填充，因而它表示对图像外部作滤波处理。而腐蚀则表示对图像内部作滤波处理。腐蚀与膨胀的另一个不同点，是膨胀满足交换律：

$$A \oplus B = B \oplus A \tag{3.45}$$

关于膨胀，还有一个等效的方程：

$$A \oplus B = \bigcup \{A + b : b \in B\} \tag{3.46}$$

因而，膨胀可以通过相对结构元素的所有点平移输入图像，然后计算其并集得到，因为膨胀满足交换律，方程还可写成：

$$A \oplus B = \bigcup \{B + a : a \in A\} \tag{3.47}$$

方程(3.47)对于分析膨胀的性质非常有用。但是，由于要对一幅输入图像的所有点作平移运算，因而计算量很大。与此相对，方程(3.46)仅需对结构元素中的所有点作平移，故运算量要小。

3、击中击不中变换(HMT)

在图像分析中，同时探测图像的内部和外部，而不仅仅局限于探测图像的内部或外部，对于研究图像中物体与背景之间的关系，往往会得到很好的结果。击中击不中变换即可达到此目的。

当利用结构元素腐蚀一幅图像时，腐蚀的过程相当于可以填入结构元素的位置作标记的过程。虽然标记点取决于原点在结构元素的相对位置，但输出图像的形状则与此相关。这是因为改变原点的位置，仅仅会导致输出结果发生平移。同样的结论也适合于腐蚀的对偶运算——膨胀。膨胀时对图像补集作腐蚀运算所得到结果的补集。

击中击不中变换在一次运算中同时可以捕获到内部标记。击中击不中需要两个结构元素 E 和 F ，这两个结构元素被做为一个结构元素对 $B = (E, F)$ ，一个探测图像内部，另一个探测图像外部，其定义为：

$$A * B = (A \Theta E) \bigcap (A^c \Theta F) \tag{3.48}$$

当且仅当 E 平移到某一点时可填入 A 的内部， F 平移到该点时可填入 A 的外部时，该点才在击中击不中变换后输出中。显然， E 和 F 应当是不相连

接的，即 $E \bigcap F = \phi$，否则，便不可能存在两个结构元素同时填入的情况。因为击中击不中变换是通过将结构元素填入图像及其补集完成运算的，故它通过结构元素对（简称结构对）探测图像和其补集之间的关系。根据腐蚀的定义式(3.42)，式(3.48)还可表示为：

$$A * B = \left\{ x : E + x \subset A; F + x \subset A^c \right\} \tag{3.49}$$

若 F 取空集，条件恒得到满足，则(3.49)变为：

$$A * B = \left\{ x : E + x \subset A \right\} = A \Theta B \tag{3.50}$$

故腐蚀可以看作击中击不中的一个特例。

4、细化(Thinning)

集合 A 被结构元素的细化用 $A \otimes B$ 表示，根据击中(hit)（或击不中 miss）变换定义：

$$\begin{aligned} A \otimes B &= A - (A * B) \\ &= A \bigcap (A * B)^c \end{aligned} \tag{3.51}$$

对称细化 A 的一个更有用的表达是基于结构元素序列：

$$\left\{ B \right\} = \left\{ B^1, B^2, \cdots, B^n \right\} \tag{3.52}$$

其中 B^i 是 B^{i-1} 的旋转。根据这个概念，本文现在定义被一个结构元素序列的细化为

$$A \otimes \left\{ B \right\} = ((\cdots((A \otimes B^1) \otimes B^2) \cdots) \otimes B^n) \tag{3.53}$$

换句话说，这个过程是用 B^1 细化 A，然后用 B^2 细化前一步细化的结果等等，直到 A 被 B^n 细化。整个过程重复进行到没有进一步的变化发生为止。

3.7.3 处理及结果分析

图像经阈值分割后，进一步的处理需要完成以下几项工作：

(1) 连接断续的道路边缘；

(2) 尽量消除道路内部的噪声点；

(3) 使道路边缘为单像素边缘，即，提取道路边缘的骨架，便于进一步的识别。

本文采用对阈值分割后的图像进行四次膨胀运算和两次腐蚀运算的方法来连接断续的道路边缘，然后应用细化算法提取道路边缘骨架，处理结果如下图所示。

1、膨胀运算一次

进行膨胀运算一次的结果图 3.16 所示：

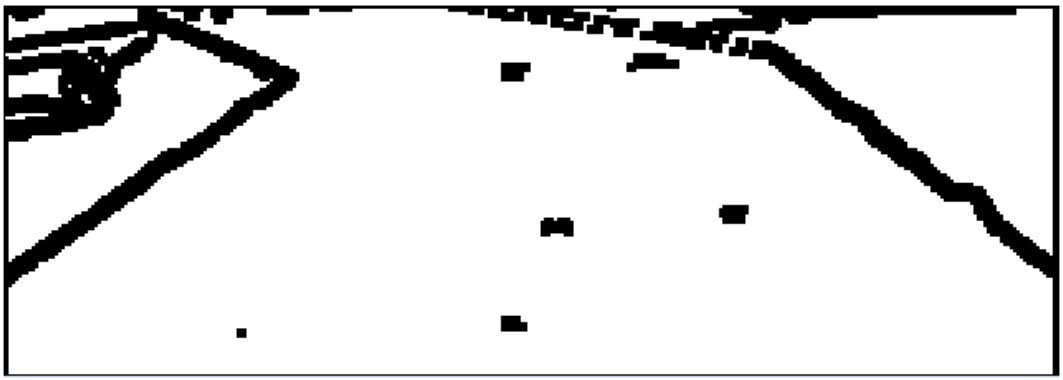

图 3.16 膨胀 1 次处理结果图

2、膨胀运算二次

进行膨胀运算二次的结果如图 3.17 所示：

图 3.17 膨胀 2 次处理结果图

3、膨胀运算三次

进行膨胀运算三次的结果如图 3.18 所示：

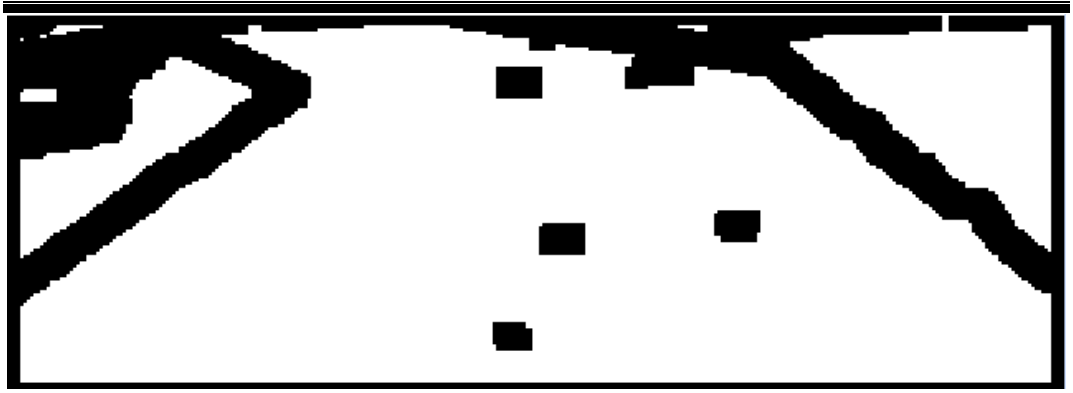

图 3.18 膨胀 3 次处理结果图

4、膨胀运算四次

进行膨胀运算四次的结果如图 3.19 所示：

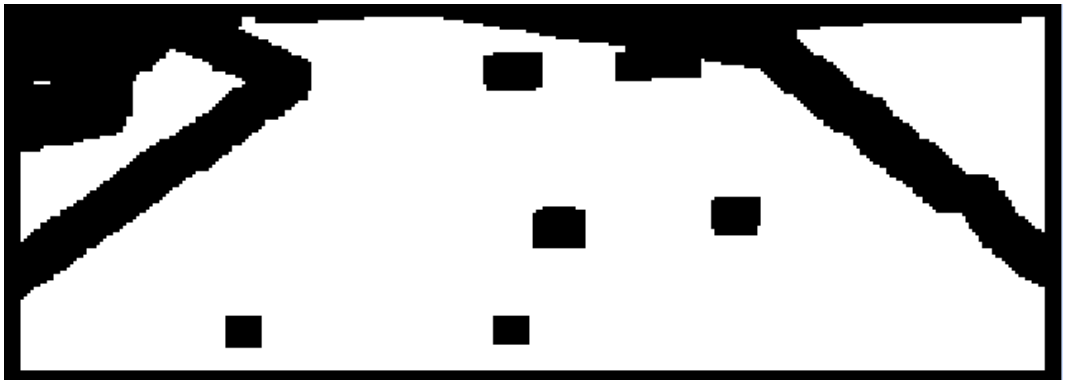

图 3.19 膨胀 4 次处理结果图

5、腐蚀运算一次

进行腐蚀运算一次的结果如图 3.20 所示：

图 3.20 腐蚀 1 次处理结果图

6、腐蚀运算二次

进行腐蚀运算二次的结果如图 3.21 所示：

图 3.21 腐蚀 2 次处理结果图

7、细化

进行细化运算的结果如图 3.22 所示：

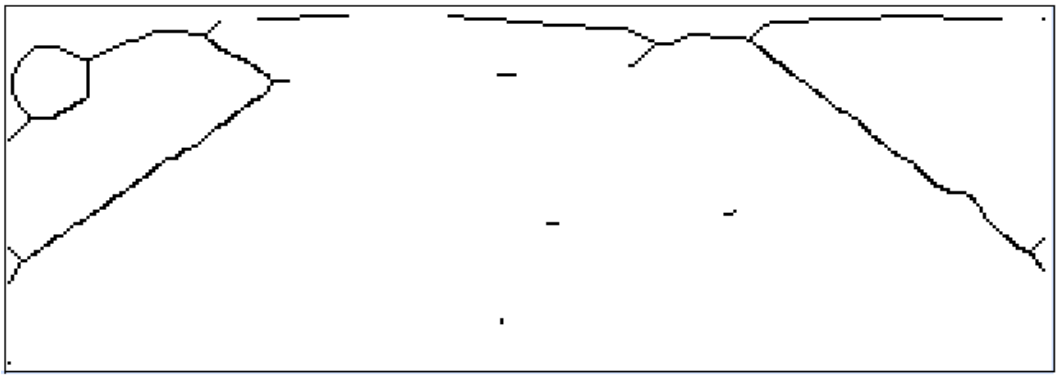

图 3.22 细化处理结果图

从处理结果可以看出道路边缘连续，且为单像素，道路边缘骨架基本被提取出来，如图 3.22 中白线所示。

3.8 本章小结

本章运用了数字图像处理学中的若干基本算法对图像进行了一系列处理。处理的结果是将待识别目标从图像背景中显现出来。然而，处理后将道路边缘提取出来的同时，也将背景中的噪声提取了出来。而本文需要图像中只有道路边缘。因此，工作还远远没有结束，本文还需要进一步的目标识别操作。

第 4 章 智能车前方视野目标识别

4.1 引言

摄像头采集到的图像经过图像处理后，背景和目标的边缘同时被提取出来了。为了从众多的边缘中仅仅提取出道路边缘，还需要进一步的识别工作。另外，本文还希望在车辆行走过程中遇到交通灯时能正确地反应，停车或继续行驶。以上即是这一章本文需要完成的内容。

4.2 道路边缘识别

4.2.1 基于道路形状的搜索算法

本算法的设计完全是基于有人驾驶时人眼对道路的感知和识别原理。人在开车时，所观察到的道路图像有以下几个特点：

①道路区域在视野中呈现为梯形，左右道路边缘在远方交于虚拟的一点；

②当这个虚拟点一直位于视野图像中间处时，车在沿道路中心行驶。若该点在视野图像左半平面，则车身偏右，反之，车身偏左；

③道路边界在时间和空间上具有连续性，一般不发生突变。因此可以通过已识别出的近处的道路边缘推知未识别的远处的道路边缘。

基于以上几点，搜索道路边缘的算法如下：

1、确定起始点

本文分三种情况来讨论智能车前方视野中的道路图像。第一种，道路右边缘落在 x 轴上，此时车辆偏右，如图所示；第二种，道路左边缘落在 x 轴上，此时车辆偏左，如图所示；第三种，道路的左、右边缘均未落在 x 轴上，此时车辆不偏离或偏离程度较小，如图 4.1 所示。

本文主要处理第三种情况下道路边缘的识别，即道路的左右边缘均未落在 x 轴上，车辆偏离不大的情况。对于第一种和第二种情况，由人的驾驶常识可知，车辆应该分别大幅度向左转弯和向右转弯，直至达到第三种情况的

状态再进行处理。

图 4.1 道路图像视野中起始点的三种情况

假设：离摄像头较近处道路的内部无噪声点。这个假设是符合常识的，因为在距摄像头较近的位置通常也是驾驶时的视觉盲点，假设离车较近的前方无噪声点，实际上等于假设自己的车行驶在道路上且偏离中间位置不多，这一点是可以通过之前的目标识别或是实地下车查实保证的。因此，所做的假设是符合情理的，且未对车辆的自动识别造成过多的干预。

同时搜索图像最左边一列（横坐标为 0）和最右边一列（横坐标为 lWidth2，其是一个常量 320），搜索方向为从图像最底一行向上搜索，如图中红色箭头所示。直到左右两边分别找到第 1 个黑点，搜索结束。因为已经假设离摄像头较近处道路的内部无噪声点，所以搜索到的第 1 个黑点即为起始点。

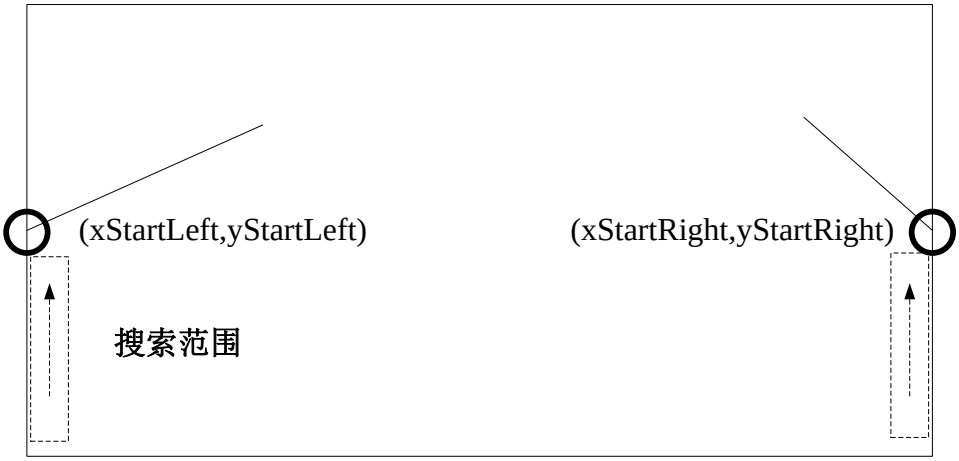

图 4.2 确定起始点示意图

2、搜索边缘

接下来本文从两边的起始点开始分别搜索道路的左、右边界。左边界的搜索方法同右边界，下面先以左边界为例。

纵向搜索方向为从起始点向上，针对每一行，搜索方向为从道路边缘向中间，初始左边缘点即为上一步搜索到的左起始点，将左起始点与图像平面中线间的垂直距离距离赋予变量 width1。若从左边缘点向中间的搜索过程中，又分以下几种情况：

(1) 如果在距离左起始点 15 个像素内搜到一个黑点，则认为该点是边缘点，算出该点距图像平面中线的距离，并将该距离值赋予 width1，下一次的搜索长度就不超过 width1。

(2) 如果搜索整行未见一个黑点，则结束该行搜索，开始下一纵行的搜索。

(3) 虽然搜索到黑点，但在距离左起始点 15 个像素外，则结束该行搜索，开始下一纵行的搜索。

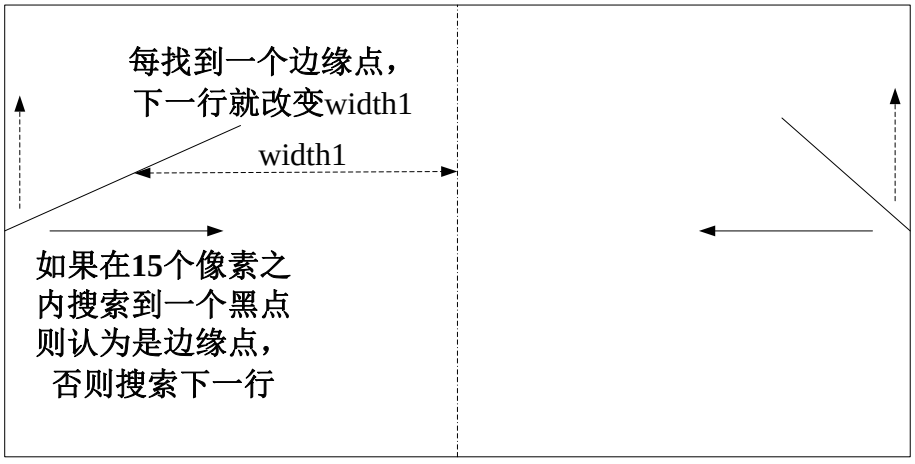

图 4.3 搜索边缘示意图

3、确定中止点

设计道路在视野范围内可能有一个转弯，所以接下来还需要设计相应得算法，使智能车能检测到这个转弯并确定其位置，进而能在转弯处正确地转向。道路设计成分段直线段，按照上述搜索道路边缘的算法搜索，转弯处必

是转弯侧的道路边缘再也被搜索不到的起始地方。按纵坐标的方向搜索图像，如果从某一行开始，连续 10 行均未搜索到一个黑点，那么起始的那行就是转弯处，且该起始行位于道路的哪一侧，则弯道就是偏向哪一侧。

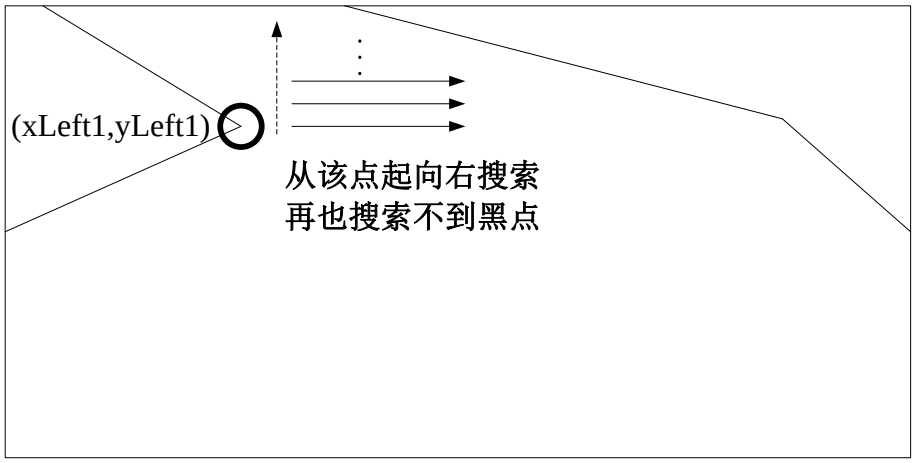

图 4.4 确定中止点示意图

4.2.2 程序流程图

上述搜索过程的程序流程图 4.5 所示。

4.2.3 识别结果及分析

按照以上的搜索流程，对图 3.22 进行处理，识别的结果如 4.6 所示：

图 4.6 道路边缘识别结果图

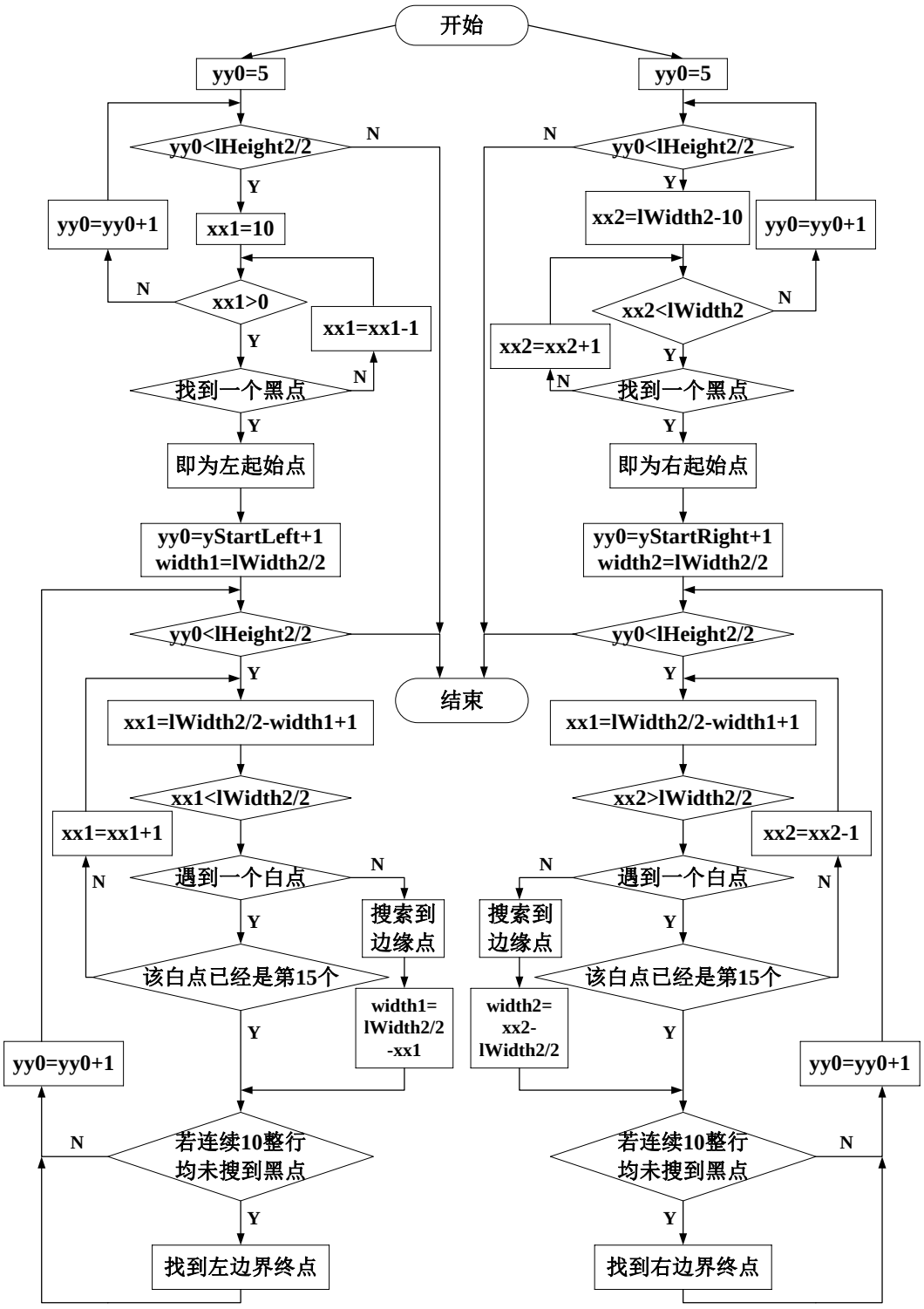

图 4.5 搜索过程程序流程图

4.3 红绿灯识别

4.3.1 模板匹配算法

匹配模型最早是针对机器的模式识别而提出来的，后来被用来解释人的模式识别。它的核心思想是认为在人的长时记忆中，贮存着许多各式各样的过去在生活中形成的外部模式的袖珍复本。这些袖珍复本即称作模板(Template)，它们与外部的模式有一对一的对应关系；当一个刺激作用于人的感官时，刺激信息得到编码并与已贮存的各种模板进行比较，然后作出决定，看哪一个模板与刺激有最佳的匹配，就把这个刺激确认为与那个模板相同。这样，模式就得到识别了。由于每个模板都与一定的意义及其他的信息相联系，受到识别的模式便得到解释或其他的加工。例如，当我们看一个字母 A，视网膜接收的信息便传到大脑，刺激信息在脑中得到相应的编码，并与记忆中贮存的各式各样的模板进行比较；通过决策过程判定它与模板 A 有最佳的匹配，于是字母 A 就得到识别；而且我们还可以知道，它是英文字母表中的第一个字母，或是考试得到的最好的分数等等。由此可见，模式识别是一个一系列连续阶段的信息加工过程。

众所周知，自动目标识别算法可以分成两大类：特征匹配法和相关匹配法。特征匹配法是通过比较标准图像目标与实时图像目标的特征来实现目标识别，它对预处理和特征提取有较高的要求，比较适合于目标特征明显、噪声较小的场合；相关匹配法通过计算实时图与参考图之间的相关测度，根据最大相关值所在位置确定实时图中目标的位置。相关匹配法具有很强的噪声抑制能力，对目标的知识要求较少，且计算形式简单、易编程实现，因而一直受到重视。

研究在一幅图中是否存在某种已知模板图像，例如在图 4.7 中，需要寻找一下有无三角形的图像。若在被搜索图中有待寻的目标，且同模板有一样的尺寸和方向，则通过一定的算法可以在图中找到目标，确定其坐标位置。它的基本原则就是通过相关函数计算来找到它以及被搜索图的坐标位置[12]。

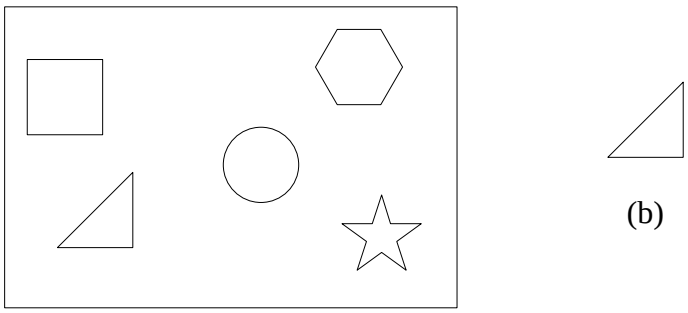

图 4.7 模板(b)与被搜索图(a)

设模板 T 叠放在搜索图 S 上平移，模板覆盖下的那块搜索图叫做子图 $S^{i,j}$，i, j 为这块子图的左上角像点在 S 图中的坐标，叫参考点，不难从图 4.8 中看出，i 和 j 的取值范围为

$$1 \le i, j \le N - M + 1$$

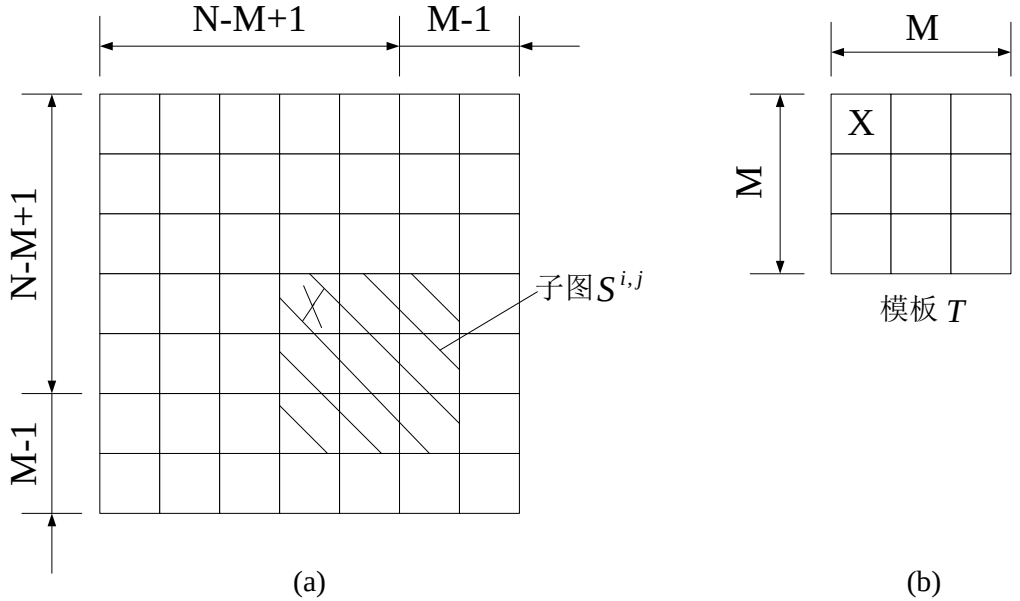

图 4.8 模板(b)及其搜索图(a)

现在可以比较 T 和 $S^{i,j}$ 的内容，若两者一致，则 T 和 $S^{i,j}$ 之差为零。所以可以用下列两种测度之一来衡量 T 和 $S^{i,j}$ 的相似程度：

$$D(i, j) = \sum_{m=1}^{M}\sum_{n=1}^{M}[S^{i,j}(m,n) - T(m,n)]^2 \qquad (4.1)$$

或

$$D(i,j)=\sum_{m=1}^{M}\sum_{n=1}^{M}\left|S^{i,j}(m,n)-T(m,n)\right| \tag{4.2}$$

若展开式(4.2)，则有

$$D(i,j)=\sum_{m=1}^{M}\sum_{n=1}^{M}[S^{i,j}(m,n)]^2-2\sum_{m=1}^{M}\sum_{n=1}^{M}S^{i,j}(m,n)\cdot T(m,n)+\sum_{m=1}^{M}\sum_{n=1}^{M}[T(m,n)]^2 \tag{4.3}$$

式右边第三项表示模板的总能量，是一个常数，与 (i,j) 无关，第一项是模板覆盖下的那块图像子图的能量，它随 (i,j) 位置而缓慢改变，第二项是子图像和模板的互相关，随 (i,j) 而改变，T 和 $S^{i,j}$ 匹配时这项取值最大，因此可用下列相关函数作相似性测度：

$$R(i,j)=\frac{D\text{的第二项}}{D\text{的第一项}}=\frac{\sum_{m=1}^{M}\sum_{n=1}^{M}S^{i,j}(m,n)\cdot T(m,n)}{\sum_{m=1}^{M}\sum_{n=1}^{M}[S^{i,j}(m,n)]^2} \tag{4.4}$$

或归一化为

$$R(i,j)=\frac{\sum_{m=1}^{M}\sum_{n=1}^{M}S^{i,j}(m,n)\cdot T(m,n)}{(\sum_{m=1}^{M}\sum_{n=1}^{M}[S^{i,j}(m,n)]^2)^{1/2}(\sum_{m=1}^{M}\sum_{n=1}^{M}[T(m,n)]^2)^{1/2}} \tag{4.5}$$

4.3.2 改进的算法

将交通灯作为待搜索的目标，搜索范围设定为摄像头采集的图像的上半部分中的某一区域。初步尝试用模板匹配法来做，将红灯和绿灯的图像事先存成模板，研究在当前采集的图像中是否存在这两种模板图像。原模板匹配法是将被搜索图与模板之间做相关计算，若在被搜索图中有待寻的目标，且同模板有一样的尺寸和方向，则找到它在被搜索图中的位置。在这里暂不要求找到目标的位置，只要确定目标存在与否即可。因此在原模板匹配法的基础上稍加改进，形成识别交通灯的算法。该算法分别检测目标图像指定位置

与模板图像中 R、G、B 值的相关系数，当检测到三个分量相关系数均超过一定阈值（例如 95%），则认为检测到了该交通灯。

4.3.3 识别结果与分析

为了提高效率、缩短程序的运行时间，可假定一个交通灯所在位置的先验知识，这是符合实际情况的。在这里指定一个区域为交通灯尽可能出现的区域：以视频采集窗口中图像左下角为坐标原点，坐标(0,160)，(0,240)，(40,160)和(40,240)所围的矩形区域。

如图所示，在红灯亮之前，指定区域与红灯模板的红、绿、蓝三个分量的最大相关系数分别约为：0.844135726，0.827051328 和 0.755116384；而红灯亮后，指定区域与红灯模板的红、绿、蓝三个分量的最大相关系数分别约为：0.992848181，0.981594339 和 0.975644566，相关程度均为 97%以上。据此可以认为识别出了红灯。

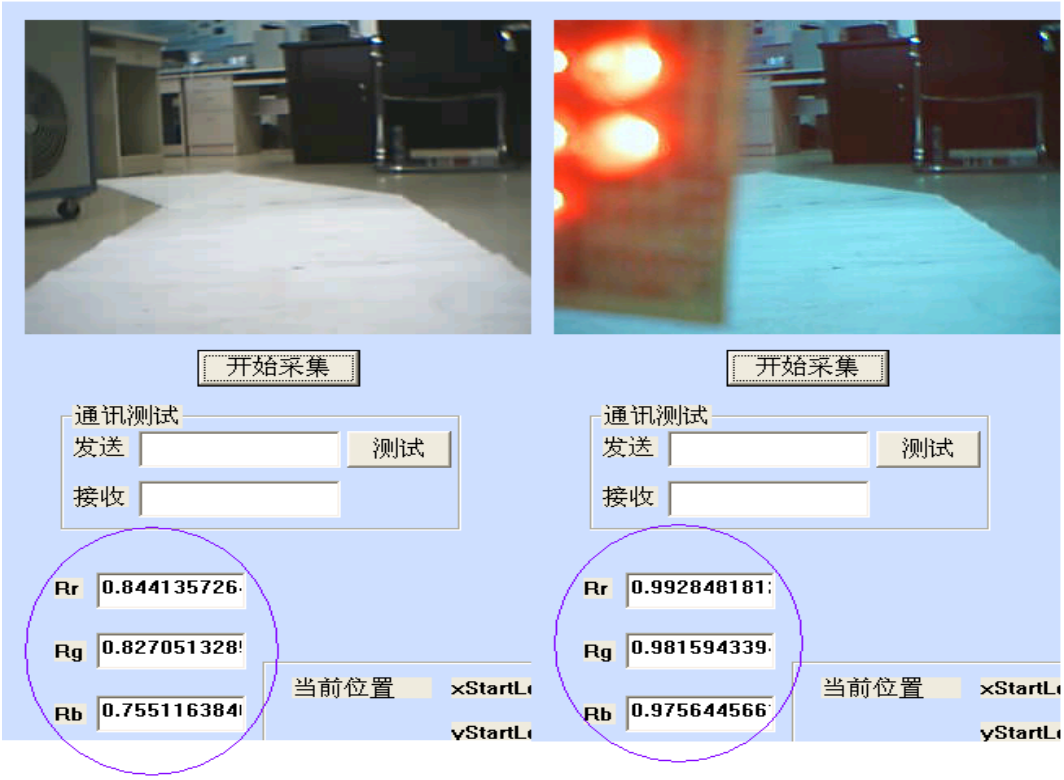

图 4.9 红灯识别结果图

该识别算法需要系统反应的时间为 1.5s，基本满足识别红绿灯的速度要求。且红灯没亮与亮之后的反差较明显，红灯亮后图像指定区域与红灯模板图像的相关性高达 97%以上，如果设定 97%为相关性阈值，大于这个阈值则提示系统红灯亮了，相应的硬件实验可以证明确实系统接到了红灯亮的指示，从而控制小车停止。

4.4 本章小结

本章系统介绍了目标识别的相关理论和方法。针对道路边缘的识别，本文应用了基于道路形状的搜索算法；对于交通灯的识别，本文应用了改进的模板匹配算法。这些算法均通过了实际系统的测试，并证明为有效的。

第 5 章 智能车系统设计与调试

5.1 引言

本文首先进行了智能车辆视觉系统的设计分析，并针对试验车进行了视觉系统的相关参数的选择，建立了车辆的单目视觉系统模型。随后本文对智能车辆图像处理方法进行了相关研究，并针对道路图像提出了实际处理效果及实时性较好的图像平滑滤波处理、边缘增强处理以及阈值分割处理和数学形态学处理方法。在此基础上，本文还介绍了基于道路形状的道路边缘识别与交通灯的识别方法。为了验证视觉系统的性能与设计的合理性和道路图像处理效果，以及检验目标识别算法的有效性，接下来本文将介绍智能车的控制算法及其软、硬件实现方法，并对实际系统进行调试。

5.2 软件界面设计与调试

5.2.1 编程语言选择及界面设计要求

世界上智能车辆系统中，绝大部分都采用 C、C++或 Visual C++语言来进行软件系统的设计。C 语言功能强大，可以直接实现对计算机底层进行控制。在 Windows 环境下，近年来的 Visual C++既保持了原有 C 语言的功能，又在程序界面、系统接口等的支持下提供了大量的源代码，大大提高了开发速度，降低了开发难度。本设计就采用了 Visual C++语言完成了上位机程序设计。在程序设计中遵循以下原则：

(1) 可靠性。可靠性也指软件的鲁棒性，它要求软件必须能够可靠的完成功能，并且可以随时接收人的控制命令。本文设计中，可以随时通过计算机键盘对车辆进行控制。

(2) 可视性。这是对人机界面的要求，要求软件系统要具有友好界面。本文设计的系统可十分方便地监视系统的运行状态，如图 5.1 为系统运行时的程序界面。可以看到，系统将算法识别出的道路边界以清晰的直线显示到

道路图像上，并且将算法识别过程中的其它信息，如道路边界起止点、车辆偏离角度、以及交通灯的识别情况等以数字形式显示出来，从而使设计者能够随时掌握系统的运行情况。

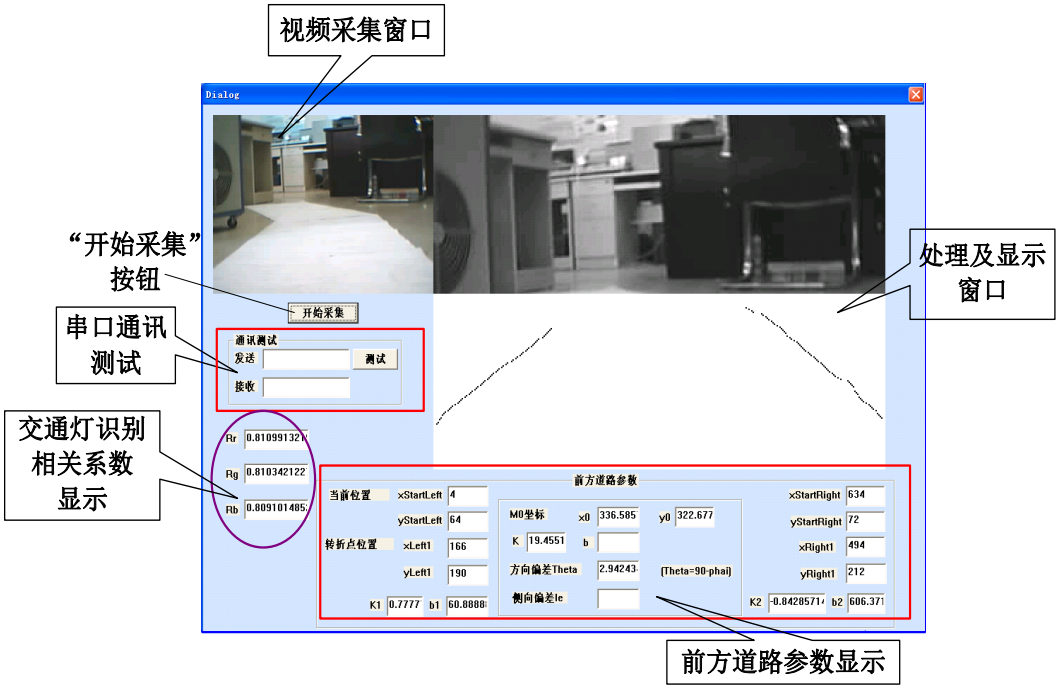

图 5.1 软件界面

5.2.2 软件界面组成及功能

软件界面由以下模块组成，现将它们的功能介绍如下：

1、视频采集窗口

将摄像头采集到的视频实时显示出来。该视频图像的大小为320像素×240像素，位深度为 24 位。

2、处理及显示窗口

将摄像头采集到的图像经过图像处理和目标识别后再次显示出来，其中处理和识别的主要是图像的下半部分区域。

3、“开始采集”按钮

点击“开始采集”按钮后，计时器开始计时，启动整个采集和处理程序。

4、“通讯测试”部分

该部分是用来测试串口通讯的有效性的。在点击“开始采集”按钮启动主程序之前可以先手动测试串口通讯部分是否有效，在发送编辑框中输入一个字符，点击“发送”后，查看接收编辑框内是否有同样的字符显示，若接收字符与发送字符相同，则证明串口通讯有效。

5、前方道路参数部分

在智能车的实时处理系统中，前方道路的各项参数均在这组编辑框中实时地显示出来，它们包括：左右两边道路边缘的起止点(xStartLeft,yStartLeft)、(xStartRight,yStartRight)、(xLeft1,yLeft1)和(xRight1,yRight1)，由此计算出的左右道路边缘所在直线的方程以及它们相交点的坐标，还有当前车辆偏离角θ，如图 5.2 所示。在接下来的控制部分中即可以通过控制这个偏离角θ使其为 0 来保证车辆行驶在道路的中央。

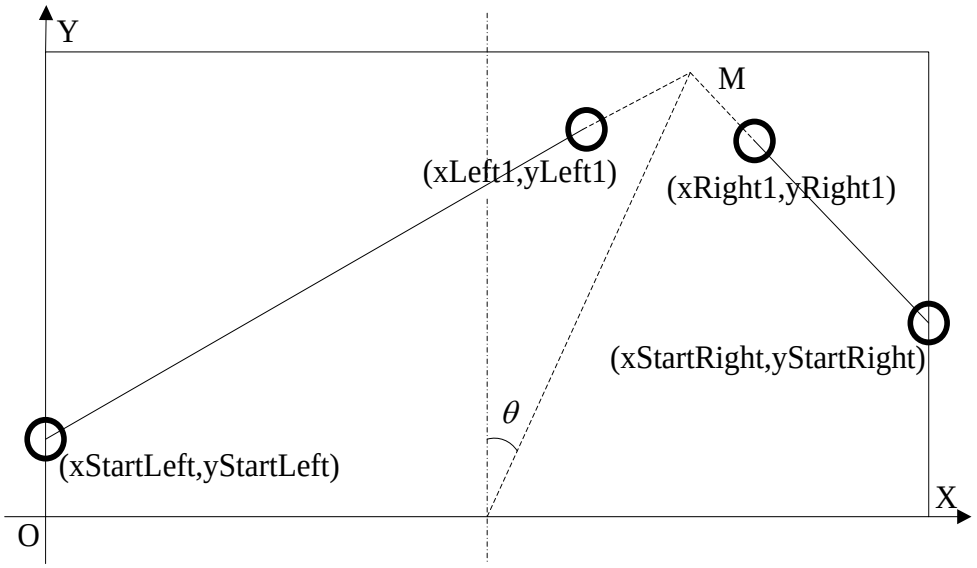

图 5.2 图像平面中路径关系

5.3 硬件系统设计与调试

5.3.1 单片机控制系统

1、AT89S51 最小系统

本控制系统的难点在于控制舵机转向。可以使用 FPGA、模拟电路、单片机来产生舵机的控制信号，但 FPGA 成本高且电路复杂。对于脉宽调制信号的脉宽变换，常用的一种方法是采用调制信号获取有源滤波后的直流电压，但是需要 50Hz（周期是 20ms）的信号，这对运放器件的选择有较高要求，从电路体积和功耗考虑也不易采用。而使用单片机作为舵机的控制单元，由于单片机系统是一个数字系统，其控制信号的变化完全依靠硬件计数，所以受外界干扰较小，整个系统工作可靠。

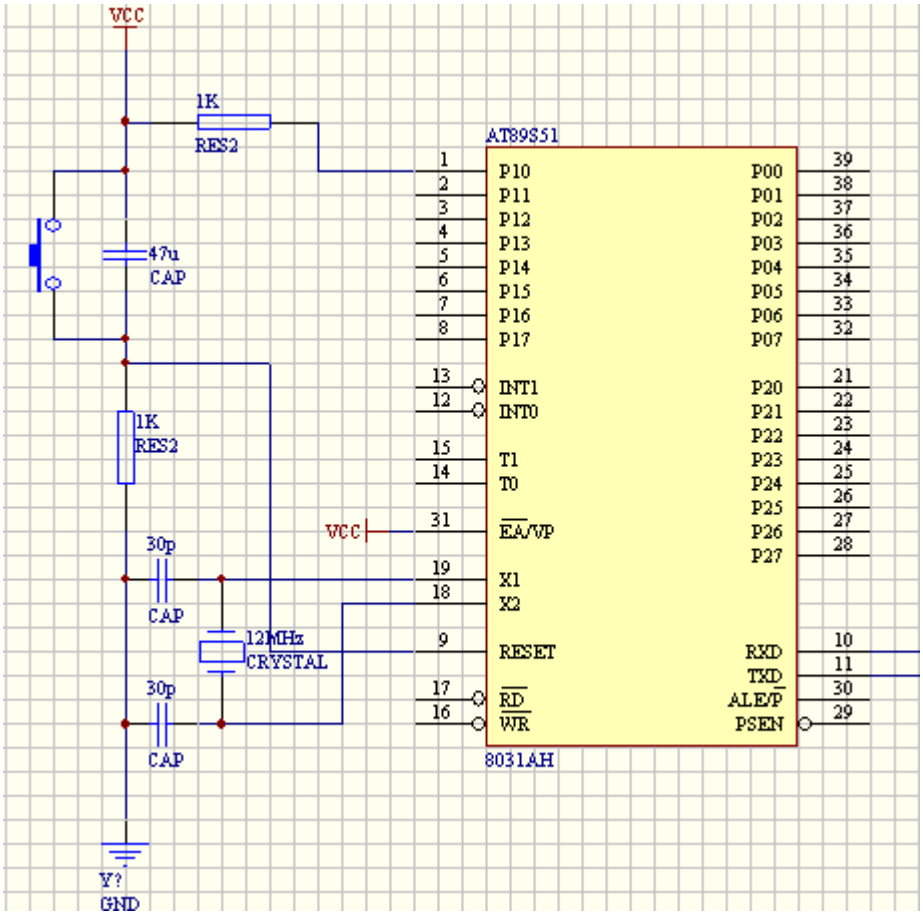

图 5.3 AT89S51 最小系统

2、MAX232 电平转换电路

在智能车系统中，上位机 PC 机与下位机单片机需要实现串口通讯。由于 PC 机端的 RS-232 电平与单片机端 TTL 电平的不匹配，故应注意电平转

换。本文采用了一种单电源供电的低功耗 RS-232 芯片 MAX232 进行电平转换。电平转换电路图如图 5.4 所示。

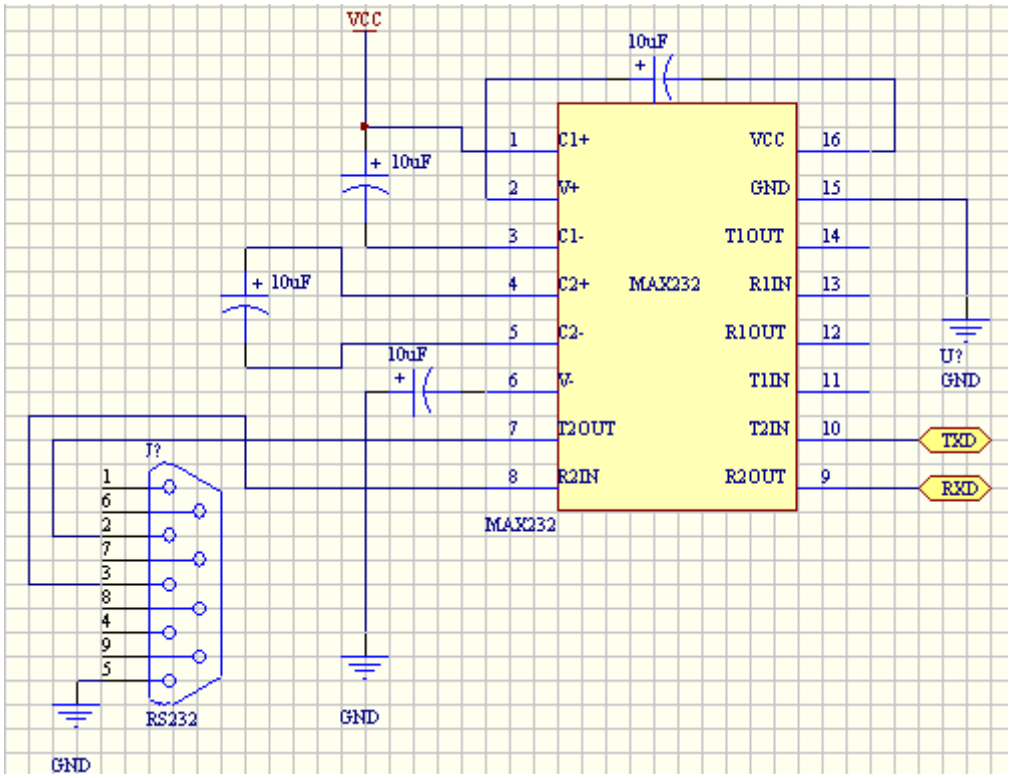

图 5.4 MAX232 电平转换电路

3、7805 稳压电路

7805 稳压电路负责为单片机系统提供标准且稳定的 5V 电源，其电路图如图 5.5 所示：

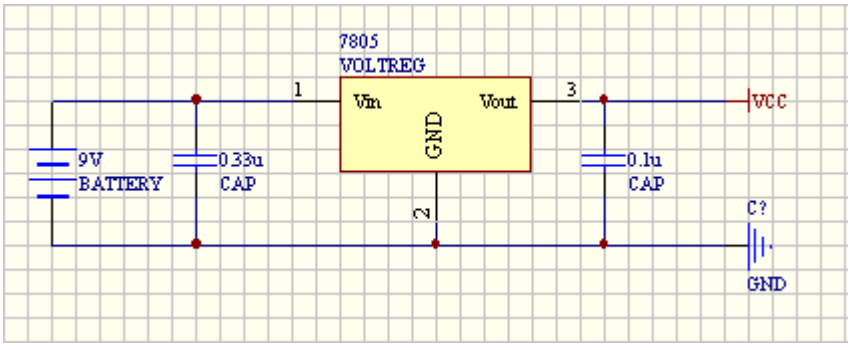

图 5.5 7805 稳压电路

5.3.2 转向控制系统

本设计采用舵机来实现智能车的转向控制。舵机是一种位置伺服的驱动器，适用于那些需要角度不断变化并可以保持的控制系统，其实物图与电路连接图如图 5.6 和图 5.7 所示。

图 5.6 舵机实物图

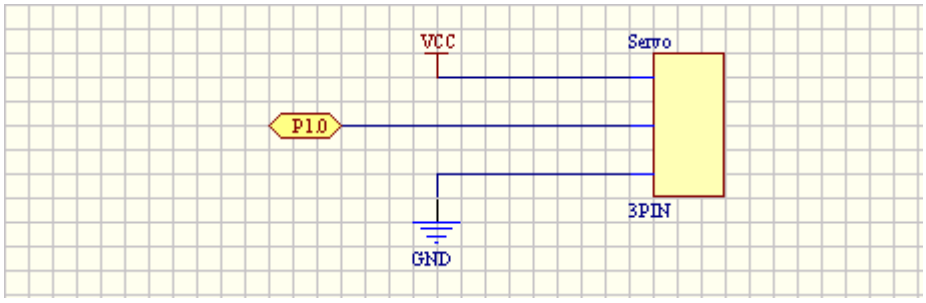

图 5.7 舵机电路连接图

舵机的工作原理是：控制信号由接收机的通道进入信号调制芯片，获得直流偏置电压。它内部有一个基准电路，产生周期为20*ms*，宽度为1.5*ms*的基准信号，将获得的直流偏置电压与电位器的电压比较，获得电压差输出。最后，电压差的正负输出到电机驱动芯片决定电机的正反转。当电机转速一定时，通过级联减速齿轮带动电位器旋转，使得电压差为零，电机停止转动。舵机的控制信号是 PWM 信号，利用占空比的变化改变舵机的位置。一般舵机的控制要求如图 5.8 所示。

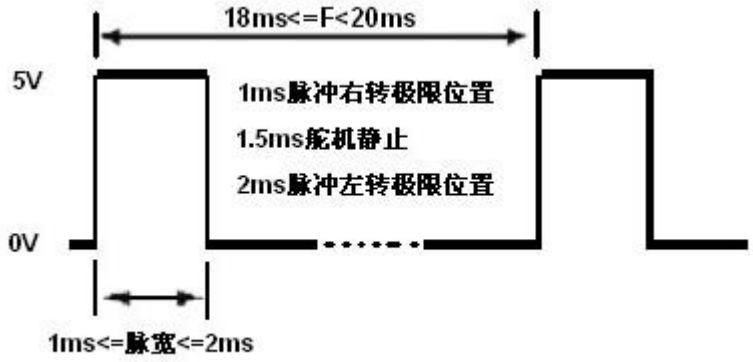

图 5.8 舵机控制要求

51 系列单片机无 PWM 输出功能，可以采用定时器配合软件的方法输出。对精度要求不高的场合，非常实用。参考程序见附录 A。

5.3.3 动力系统

该智能车需要动力装置来推动其前行，并且在遭遇红灯时可以停车。因为整个过程小车只需要完成行进和停止操作，不需要改变行驶速度，因此对智能车的整个控制过程非常简单，采用普通减速电机即可以胜任。电路连接图如图 5.9 所示：

5.3.4 结果与分析

经过测试，上位机系统能根据目前采集到的图像向下位机发出正确的转向和行止指令，下位机接收到指令后，能够正确控制舵机的转角以及电机的

转动或停止。整个系统设计证明是有效的，但还需要进一步的调试来提高系统运行的精度和稳定性。

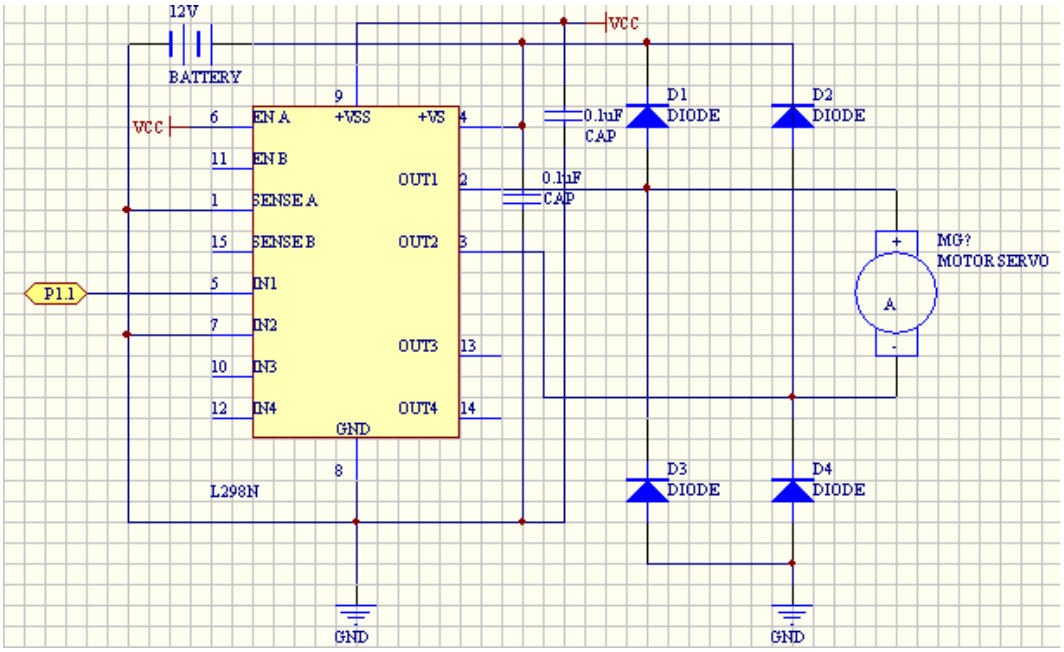

图 5.9 电机驱动电路

5.4 本章小结

本章首先选择 VC++6.0 环境完成了智能车的软件系统设计，包括界面的设计和智能车控制算法的设计；随后本章又通过简单的电路设计搭建了智能车的硬件系统。通过实际测试，证明整个系统运行有效。

结　论

智能车辆是当今世界车辆工程研究领域的研究前沿和热点，而智能车辆视觉导航的研究是目前智能车辆研究的重要课题，智能车辆视觉系统及其关键技术的研究方兴未艾。本文着重设计智能车模拟自动驾驶软件系统，并通过硬件测试验证软件系统运行的有效性。论文主要研究内容包括：

1、对智能车的视觉系统进行了相关研究，根据要求选择摄像头，并应用视频采集软件包 VFW 进行视频采集和处理；

2、对智能车道路图像处理方法进行了相关研究。针对实际的道路图像采用了灰度化处理、中值滤波、Sobel 边缘检测、阈值分割、数学形态学处理等一系列处理方法，并通过对处理结果进行分析证明其有效；

3、对智能车目标识别进行了相关研究。编写了基于道路形状的道路搜索算法来识别道路边缘，同时应用改进的模板匹配法来识别交通灯。

4、对智能车控制系统的设计，包括被控量的选择与计算以及系统软件界面和硬件电路的设计，并通过实际测试证明了设计的有效性。

总之，随着对智能车辆研究开发的不断深入，智能车辆相关技术将日趋成熟，智能车辆的智能化程度也将越来越高，智能车辆也必将逐渐走进人们的生活，产生明显的社会和经济效益。

参考文献

[1]占强. 从科幻到现实——无人驾驶城市挑战赛[J]. 世界汽车，2007，12：90-93.

[2]蒋杰. 新型智能车辆视觉系统及图像处理技术的研究[D]. 吉林：吉林大学，2003.

[3]刘显荣. Windows平台下视频捕捉的几种实现方法[J]. 重庆科技学院学报，2006，8(2)：96-98.

[4]李江华，谢红，王晓丹. 基于VFW实时视频捕获原理与实现[J]. 应用科技，2005，32(9)：53-55.

[5]四维科技，刘祎玮. Visual C++视频/音频开发实用工程案例精选[M]. 北京：人民邮电出版社，2006.

[6]沈旭. 一种实现视频捕捉的简单方法[J]. 计算机与信息技术，2008，1：14-16.

[7]冯晓倩，宋仲康，宋慧敏. 基于VC++的视频捕捉技术[J]. 仪表技术，2003.

[8]叶家鸣. 彩色城市交通地图道路信息的识别与提取[D]. 安徽：中国科学技术大学，2003.

[9]求是科技. Visual C++数字图像处理典型算法及实现[M]. 北京：人民邮电出版社，2007.

[10]李俊山，李旭辉. 数字图像处理[M]. 北京：清华大学出版社，2007.

[11]高木干雄，下田阳久，孙卫东. 图像处理技术手册[M]. 北京：科学出版社，2007.

[12]沈庭芝，方子文. 数字图像处理及模式识别[M]. 北京：北京理工大学出版社，2005.

[13]冈萨雷斯. 数字图像处理（第二版）[M]. 北京：电子工业出版社，2007.

[14]杨淑莹. VC++图像处理程序设计[M]. 北京：清华大学出版社，北京交通大学出版社，2005.

[15]求是科技，张宏林. Visual C++数字图像模式识别技术及工程实践[M]. 北京：人民邮电出版社，2003.

[16]章毓晋. 图像处理和分析[M]. 北京：清华大学出版社，2001.

[17]章毓晋. 图像分割[M]. 北京：清华大学出版社，2001.

[18]章毓晋. 图像理解与计算机视觉[M]. 北京：清华大学出版社，2001.

[19]周长发. 精通VC++图像处理编程[M]. 北京：电子工业出版社，2004.

[20]郎锐. 数字图像处理学Visual C++实现[M]. 北京：北京希望电子出版社，2002.

[21]徐杰. 智能车辆视觉导航中道路边界识别技术的研究[D]. 吉林：吉林大学，2001.

[22]孙鑫，余安萍. VC++深入详解[M]. 北京：电子工业出版社，2006.

[23]李欣，李宏东，顾伟康，李庆中. 一种结构化道路环境中的视觉导航系统[J]. 浙江大学学报（工学版），2002，36(6)：630-633.

[24]李进，陈无畏，李碧春，王檀彬. 自动导引车视觉导航的路径设别和跟踪控制[J].

[25]沈中杰，王武宏，钟永刚. 智能交通系统中基于机器视觉的数字车辆控制[J]. 汽车工程，2003，25(5)：428-433.

[26]徐爱钧. 单片机高级语言C51 Windows环境编程与应用[M]. 北京：电子工业出版社，2001.

[27]时玮. 利用单片机PWM信号进行舵机控制[J]. 应用天地，2005，11(1)：80-82.

[28]吴化波，钱春来. 基于AT89C2051的多路舵机控制器设计[J]. 应用天地，2006，8(1)：55-58.

[29]卓晴，黄开胜，邵贝贝. 学做智能车——挑战“飞斯卡尔”杯[M]. 北京：北京航空航天大学出版社，2007.

致谢

在本文的最后，我衷心感谢莫宏伟老师。本文是在他悉心的指导下完成，莫老师教导我作科学研究要有严谨、负责的态度还要掌握科学、合理的方法以及拥有平和的心态、乐观的态度。感谢莫宏伟老师，恩师教导永不忘。

同时，也要感谢 408 教研室，以及工程训练中心的老师和学长们，在我写论文期间给我很大帮助，教给我很多知识和方法，在此提出感谢！

感谢和我同组的张帆和张卓然同学，在做毕业设计期间我们共同交流和探讨，让我受益匪浅。同时也要感谢在毕业设计期间给予我莫大帮助的李湛、刘朕以及张衍儒同学，他们富有创造性的设计思路给了我很大启发。

感谢我的父母，感谢他们的无私关爱和鼓励。

在这短短的几个月的毕业设计中我学到了很多，不仅是知识的积累和运用，更是严谨求实的治学态度和挑战困难的精神。这期间所学的一切是我一生都不可忽视的宝贵财富。在以后的人生道路上，我会时时以这段经历来勉励自己，继续攀登事业的高峰。