南京理工大学
NANJING UNIVERSITY OF SCIENCE & TECHNOLOGY

毕业设计说明书

作　　者：刘驰　　学　号：912106840316

学　　院：计算机科学与工程学院

专业(方向)：网络工程

题　　目：车牌识别技术在嵌入式系统上的开发与实现

指导者：孙晋　　副教授
(姓　名)　　(专业技术职务)

评阅者：
(姓　名)　　(专业技术职务)

2016 年　6　月

声　明

我声明，本毕业设计说明书及其研究工作和所取得的成果是本人在导师的指导下独立完成的。研究过程中利用的所有资料均已在参考文献中列出，其他人员或机构对本毕业设计工作做出的贡献也已在致谢部分说明。

本毕业设计说明书不涉及任何秘密，南京理工大学有权保存其电子和纸质文档，可以借阅或网上公布其部分或全部内容，可以向有关部门或机构送交并授权保存、借阅或网上公布其部分或全部内容。

学生签名：

年　月　日

指导教师签名：

年　月　日

毕业设计说明书中文摘要

本文以车牌识别技术在 ARM 嵌入式系统上的开发与实现作为研究内容，通过在嵌入式系统上移植车牌识别算法以实现车辆图片的采集、识别与存储功能。本文首先介绍了智能交通的背景和以及车牌识别的研究现状，然后对车牌识别的相关基础知识做出了介绍，接着详细阐述了车牌识别关键算法中的主要功能模块，包括图像的采集、图像的预处理、车牌的定位、字符的分割、字符的识别、结果处理等。此外，在系统设计方面进行了嵌入式 Linux 系统的搭建、内核的裁剪和移植，在嵌入式平台上通过 OpenCV 图像处理函数库和 Qt/Embedded 开发环境完成软件程序的编写。系统测试结果表明，本课题所实现的嵌入式车牌识别系统对国内车辆的单行牌照具有较高的识别率，而对于双行车牌的识别将在后续的研究工作中进一步展开。

关键字　嵌入式系统　车牌定位　字符分割　字符识别　OpenCV

毕业设计说明书外文摘要

Title The design and implementation of a license plate recognition technique in embedded systems

Abstract

The research task of this project is the design and implementation of a license plate recognition technique on ARM embedded systems. By transplanting the license plate recognition algorithm into an embedded platform, this project implements the functionalities of vehicle image acquisition, license plate image recognition and storage functions. This thesis first introduces the background knowledge of intelligent traffic, as well as existing license plate recognition methods and products. The paper further introduces the preliminary knowledge of license plate recognition. Furthermore, this thesis presents in details the basic functional blocks of the license plate recognition algorithm, including image acquisition, image preprocessing, license plate location, character segmentation, character recognition and result processing. From the perspective of system design, this work has built an embedded Linux system by kernel trimming and program transplantation, as well as implementing the algorithm based on the OpenCV image processing function library and Qt/Embedded development environment. Verification results indicate that the designed system achieves a high recognition accuracy for Chinese single line car license plates. The research on the effective recognition of double line license plates will be conducted in further research tasks.

Keywords Embedded system License location Character segmentation Character recognition OpenCV

目 录

1 绪论.....1
1.1 研究背景.....1
1.2 国内外研究现状.....1
1.3 本课题主要工作.....2
1.4 本文结构安排.....3
2 预备知识.....4
2.1 彩色图像的基本概念.....4
2.2 OpenCV 和 QT 简介.....6
2.3 中国车牌的特点.....6
2.4 本章小结.....8
3 关键算法分析.....9
3.1 图像的预处理算法.....9
3.2 车牌定位算法.....12
3.3 字符提取算法.....15
3.4 字符识别算法.....16
3.5 本章小结.....17
4 系统实现.....18
4.1 嵌入式系统概述.....18
4.2 总体设计方案.....19
4.3 Linux 系统与动态库的移植.....22
4.4 相关函数实现.....22
4.5 相关 UI 设计.....24
4.6 识别程序的嵌入式实现.....25
4.7 本章小结.....33
结 论.....34
致 谢.....35
参 考 文 献.....36

图 2.1 车牌识别系统的基本工作流程.....4
图 2.2 中国车牌规格.....8
图 3.1 模板匹配算法.....17
图 4.1 移植程序流程.....19
图 4.2 Linux 安装.....20
图 4.3 OpenCV 安装.....21

图 4.4 整合 Qtcreator.....22
图 4.5 车牌识别系统 UI.....25
图 4.6 实验结果 1.....26
图 4.7 实验结果 1 灰度化图.....27
图 4.8 实验结果 1 边缘检测图.....27
图 4.9 实验结果 1 车牌定位图.....27
图 4.10 实验结果 1 字符 1 图.....28
图 4.11 实验结果 1 字符 2 图.....28
图 4.12 实验结果 1 字符 3 图.....28
图 4.13 实验结果 1 字符 4 图.....28
图 4.14 实验结果 1 字符 5 图.....29
图 4.15 实验结果 1 字符 6 图.....29
图 4.16 实验结果 1 字符 7 图.....29
图 4.17 实验结果 2.....30
图 4.18 实验结果 2 灰度化图.....30
图 4.19 实验结果 2 二值化图.....31
图 4.20 实验结果 2 车牌定位图.....31
图 4.21 实验结果 2 字符 1 图.....31
图 4.22 实验结果 2 字符 2 图.....31
图 4.23 实验结果 2 字符 3 图.....32
图 4.24 实验结果 2 字符 4 图.....32
图 4.25 实验结果 2 字符 5 图.....32
图 4.26 实验结果 2 字符 6 图.....32
图 4.27 实验结果 2 字符 7 图.....32

表 2.1 常见颜色的 RGB 值.....5
表 2.2 各种图像颜色深度.....5
表 2.3 中国车牌构成.....7
表 4.1 主要软件用途.....20
表 4.2 车牌识别主要函数表.....24

1 绪论

1.1 研究背景

随着人类科学技术的进步和经济的快速发展，人类对于车辆的使用越来越普遍，随之产生的交通道路拥堵的问题也越来越严重。早在 30 年前，许多发达国家就为了解决这一现实问题而提出了一个概念——智能交通系统（Intelligent Transportation Systems，ITS）。ITS 是指利用人类先进的自动控制技术、计算机技术、传感技术、网络技术、电子通讯技术，通过计算机来控制车辆、交通和道路，并向人类提供相当舒适、安全和高效的交通系统。

车牌识别（Vehicle License Plate Recognition，VLPR）系统，是上述智能交通系统的一个非常重要的组成部分。VLPR 的主要功能是使用摄像设备采集原始车辆图像，然后通过采用相应的人工智能、机器视觉、图像处理等技术，在原始图像中检测出车牌的位置并提取车牌，继而对车牌字符图像进行分割，并使用字符识别技术识别出车牌中的字母、数字以及各种国家的特定字符，最后得到车牌号码。

车牌识别系统在当下有着相当重要的意义和广泛的应用前景。其主要有以下四个方面的应用：(1). 城市交通管理和控制；(2). 收费站监控；(3). 小区入口停车场入口管理；(4). 高速公路卡口监控。这些应用能够实现车辆自动管理的功能，为交警部门高效地监管和抓捕部分违法车辆提供便利，节省大量的物力、人力和财力。除此之外，通过动态管控道路状况，可以缓解交通需求矛盾的进一步加剧，满足新时代对治安、刑侦等新形式下的业务需求[1][2]。

1.2 国内外研究现状

从上个世纪 90 年代至今，世界各国都在进行关于智能交通系统的许多研究，也在积极开展车牌识别系统的探索工作，目前取得了一些成果，并运用到了实际中。虽然业界使用了很多技术方法，但受到外界环境的干扰，比如光线变化、光照不均以及车牌自身字符模糊等原因，VLPR 系统的实际使用状况一直有待加强。

国外目前已经有了一些相对成熟的车牌识别产品，其中包括：香港的 Asia Vision Technology 公司的 VECON 产品，以色列的 See/Car System 系列，新加坡的 Optasia 公司的 VLPS 系列等等。除此之外，英国、日本、美国等一些发达国家也有了适合本国车牌的产品，并且相关的研究还在继续[5]。

1991 年，国际著名学者 C.Oliver 与 R.Mullot 等人提取了车牌和车牌的文字纹理特征，并使用机器学习方法来提取它们的共性特征，从而高效监测出原始图像中车牌文字的位置。此类方法最初用于识别集装箱上的编号，后来被引入车牌识别领域，并且也取得了很好的识别

效果[6]。Thanongsak Sirithnaphong 等人提出了应用 BP 神经网络的方法识别车牌，实验结果的准确率在 84%左右[7]。目前国外较好的车牌定位方法包括 P.R.的离散傅里叶变换的频域分析方法，还有 B. J.提出的水平线搜索的方法，一集基于局部二值化的方法[8][9]。

与国外的技术相比，国内的车牌识别技术还处于不断完善的阶段，有许多地方是通过人工操作来进行识别的。国外的先进技术在国内虽然也有很多借鉴意义，但是中国的牌照有以下一些自身特点。

（1）汉字识别难度大，车牌内容包含了数字、汉字、字母。

（2）车牌号码的颜色众多，包括红白黑，同时底色也是白蓝黑黄四种颜色。

（3）车辆种类繁多，各种车的牌照也不尽相同。

（4）牌照固定的位置层次不齐，安装位置不固定会造成定位困难。

我国目前比较成熟的产品由杭州海康威视、北京信路威、汉王公司生产。这些产品基本满足大众使用需求，同时，各大高校也在实验室中进行着相关的研究。朱伟刚等人根据图像的颜色特征，分割出图像中的车牌信息特征，并取得了很好的分割效果[10]。戚飞虎和赵雪春等人在图像颜色分割的基础上，额外增加了多级混合的检测器来实现车牌的识别[11]。张引等人利用了区域生长算子以及边缘检测算子 Color Prewitt 算子，对图像中的车牌进行有效分割[12]。张玲等人在车牌的定位算法中引入了遗传算法引，尽管处理速度稍慢，但定位效果非常好[13]。王玫等人利用车牌颜色特征的伴生性和互补性，也实现了车牌的有效定位[14]。

目前市场上存在的大部分车牌识别系统基本都依赖于 PC 机运行，而在某些特殊场合中，车牌识别功能对系统的体积、性能、实用性和性价比有着相对严格的要求。在这样的应用场景下，基于 PC 的车牌识别系统的实用价值受到很大的限制，因而凸显出本文中的将车牌识别技术与嵌入式系统相结合的重要性与必要性。设计出一套更具实用性、更加便捷精巧的嵌入式车牌识别系统，将会为车牌识别系统的发展提供更加广阔的空间。

1.3 本课题主要工作

本课题完成的主要工作为一种有效的车牌识别算法的 C++程序设计，并最终在 ARM 嵌入式平台加以实现。本课题采用以 ARM 为核心的开发板作为系统的硬件平台，通过移植必需的操作系统和软件开发库，在此嵌入式平台上实现所设计的车牌识别算法。

本课题的主要研究内容围绕以下几个方面展开：一、基于 OpenCV 开源库，使用 C++语言实现车牌识别算法中的主要功能模块，包括车牌图像预处理、车牌的定位、字符分割、字符识别等；二、在 ARM 开发板上移植 Linux 操作系统、QT 图形化界面库和 OpenCV 计算机视觉库，搭建识别算法所需的嵌入式平台；三、编写嵌入式程序对识别算法进行嵌入式平台

的移植，并实现相应的用户界面完成嵌入式车牌识别的设计。

1.4 本文结构安排

第一章 绪论主要介绍了本文的研究背景研究现状以及相关文献。

第二章 主要介绍在进行车牌处理之前那所需要知道的一些预备知识，包括图像的彩色图像的一些基本概念，图像处理时需要的一些概念，以及需要使用的库文件和图形化界面。最后还介绍了中国车牌的主要特点，以方便后续算法的实现。

第三章 本章主要介绍了车牌识别的主要算法，包括了图像的预处理算法，车牌的定位算法，字符分割的算法和字符识别的算法。

第四章 主要介绍了嵌入式平台的基本知识，系统的实现，包括搭建开发环境，安装交叉编译工具，整合 QtCreator，裁剪并移植 Linux 系统，移植识别系统。最后给出了实验结果。

2 预备知识

一般来说，一个典型的车牌识别系统[15]，包括由前端设备进行的采集图像额过程，分析仪器进行的车牌识别处理，以及后端服务器进行的相关结果的处理。工作流程大致有下面几步：采集原始图像、预处理原始图像、车牌图像的定位、分割字符、识别字符、结果处理。

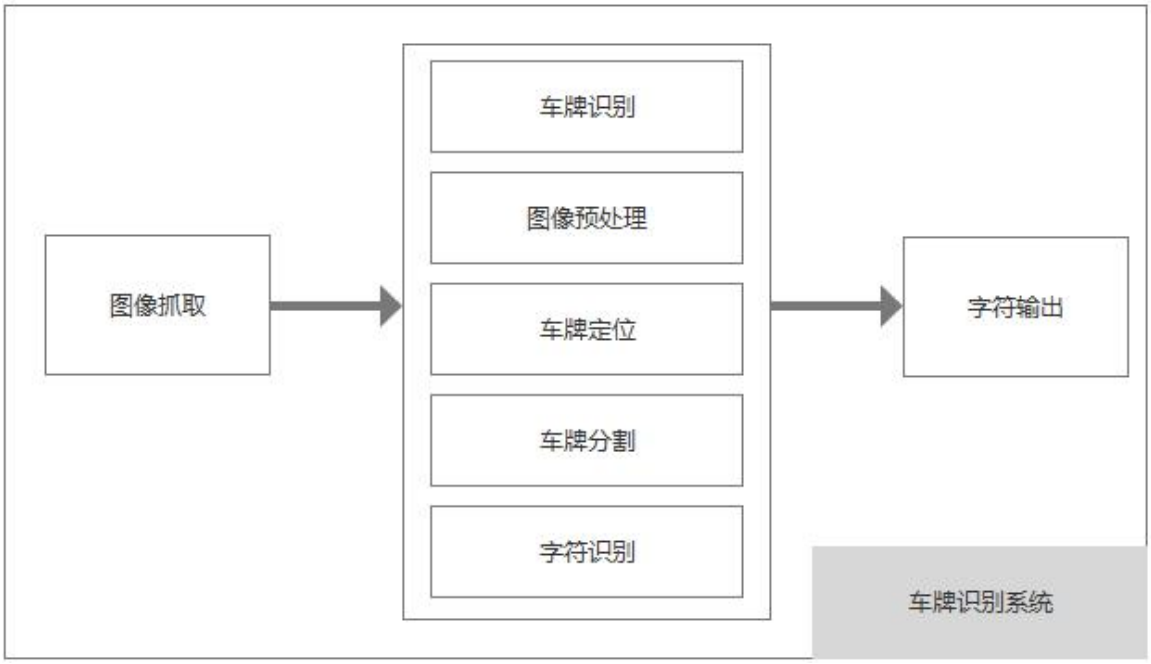

图 2.1 车牌识别系统的基本工作流程

在图像采集的过程中，由于外界的各种不确定因素的干扰，导致了图像的质量有所下降，以及后续的识别效果不佳。因此在图像识别之前，需要先对牌照图像进行预处理，以达到尽量消除噪声、抑制无用特性、突出需要识别区域的特征的作用。

车牌原始图像一般会包含车牌和车牌的背景图像，在进行下一步的字符分割之前，首先要定位到车牌的位置，并将车牌从原始图像中分离出来。如果车牌定位算法不够准确，字符分割的难度将会增加，甚至无效，从而整个系统的识别率会受到影响。

字符分割的作用是在已经得到的车牌图像中找到每个字符的图像，并进行分割，字符图像的分割有利于单个字符的的识别。字符提取的基本策略有：基于图像特征的分割策略和基于识别的分割策略。

经过字符分割后的单个字符图像大小并不一定一样，所以为了便于字符识别，需要我们将输入进来的字符图片进行统一尺寸的操作。

2.1 彩色图像的基本概念

计算机中显示的任何颜色都可以由蓝、绿、红三种颜色表示，简称为三基色。常见的 7 种颜色对应的 RGB 值如下。

表 2.1 常见颜色的 RGB 值

颜色名	R 值	G 值	B 值
红	255	0	0
绿	0	255	0
蓝	0	0	255
白	255	255	255
黑	0	0	0
青	0	255	255
紫	255	0	255
黄	255	255	0

2.1.1 颜色的三个基本属性

颜色在视觉上可以分为非彩色和彩色这两大类。有色系列或者彩色系列，是指白色以外的其他各种颜色。饱和度、色调、亮度，这是颜色的 3 个基本属性。

2.1.2 图像深度

图像深度限制了彩色图像中最多颜色数目，同时也决定了灰度图像中最大的灰度等级数。我们用三维空间表示图像的色彩，如 RGB 空间。因为色彩空间的表示方法有很多种，所以如何分配像素的图像深度，与图像所在的色彩空间也有很大关系。在我们讨论映象图像的彩色时，通常定义彩色数的方式是使用保存色彩信息所有的位数。各种位图的图像深度如下表。

表 2.2 各种图像颜色深度

位图	颜色数	图像深度
单色图	2	1 位
灰度图	256	8 位
伪彩色图	256	8 位
24 位真彩色图	1667 万	24 位

2.1.3 灰度图像

与彩色图像相似，不同在于查表后的 R、G、B 三个分量的值是相等的。除了白色和黑色，还有 254 种深浅的灰色。

2.1.4 黑白图像

黑白图像由 1 位元来表示每一个像素，颜色就只有黑色和白色。好处是占的内存小，方便计算，但是功能就相对较少。比如黑白图像不能表示渐变色。

2.2 OpenCV 和 QT 简介

OpenCV（Open Source Computer Vision, 开源计算机视觉库）是由 Intel 开发的一个提供实时图像处理的免费跨平台库。OpenCV 实际上已经成为一切计算机视觉相关处理的一个标准库工具，并且能够在大部分流行操作系统中使用。OpenCV 的应用包括识别与分割、三维和二维特征工具包、人脸识别、对象识别、图像拼接、手势识别、增强现实、运动追踪、高动态范围成像[16]。

开发 OpenCV C++应用需要以下几个先决条件：（1）对 OpenCV 头文件和库文件进行编译。OpenCV 的代码包应该使用与生产用户应用程序相同的编译器来编译。（2）一个 C++编译器，包括代码编辑器、项目管理器、调试器、修订控制系统、构建过程管理器、类检测器等。通常这些工具在同一个 IDE 中进行配置。（3）

其他辅助库：有时编写最终应用程序需要任何其他辅助库，例如统计、绘图等。

目前主流的用于编写 OpenCV C++应用的编译工具有以下几种。

（1）Microsoft Visual C++（MSVC）：与 IDE Visual Studio 的集成度很高，只在 Windows 环境下提供支持（当然也可以与其他跨平台的 IDE 集成，例如 QT Creator 或者 Eclipse）。与目前最新的 OpenCV 发布兼容的 MSVC 版本是 VC15。

（2）CNU Compiler Collection GNU GCC：这是由 GNU 项目组开发的一个跨平台的编译系统，在 Windows 开发环境下就是广为人知的 MinGW。与目前 OpenCV 发布兼容的 GNU GCC 版本为 4.8，该工具包可以与若干 IDE 一起使用，例如 Eclipse、QT Creator、Code::Blocks 等。

QT 是一个跨平台的 C++图形用户界面的应用程序框架，拥有丰富的 API 和大量的开发文档。以 QT 为基础，QT/Embedded 是面向嵌入式的 QT 版本。QT Creator 首次提供专为跨平台开发而设计的 IDE，提供了一套高效的工具用于测试和创建基于 QT 的程序，包括源代码管理、C++代码编辑器、可视化调节器、上下文感知帮助系统、项目管理工具和构建管理工具。最新的 QT 软件开发工具包包括了 QT Creator IDE、QT 库和 QT 工具，并对 Mac、Linux/X11 和 Windo 近 ws 等主流操作系统提供支持。

Qtopia 起源于 QPE（Qt Palmtop Enviroment, QT 掌上电脑环境），是构建于 QT/Embedded 之上的一系列应用程序。QT/Embedded 可以独立使用，也可以与 Qtopia 配合。

2.3 中国车牌的特点

汽车牌照在每个国家都有着不同的规定，国外早就有了比较成熟的车牌识别系统，但是

不能直接应用到中国车牌上。本文研究的重点在中国大陆的汽车车牌识别，所以下文将会阐述中国大陆的车牌使用规范。汽车牌照，是车辆行驶执照和汽车号牌的检测，我国于 1972 年颁布的法律条文《城市和公路交通管理规则》，明确用行驶证替换行车执照[3]。

在 92 形式的车牌规定中，大型民用车牌号码使用黑色字，而底色要求是黄色。小型农用车牌号码使用白色字，而底色要求是蓝色。公安系统的专用车辆的车牌号码第一位是红色汉字“警”字，随后跟黑色字，而底色要求是白色。中国境内大使馆的外籍车辆牌照号码是白色，而底色要求是黑色。驾校的教练车车牌号码是黑色，而底色要求是黄色，除此之外，车牌的第一个字是白色的“学”字。试车的车牌号码是红色，并且第一个汉字是白色的“试”字，而底色要求是白色。临时办理的车牌号码是黑色，而底色要求是白色。车辆的不用牌照号码是黑色，而底色是白色。

中国大陆区域使用的牌照，根据行政区域有着明确的划分。车牌号的首位为省或者直辖市的简称。第二位是字母，从 A 开始计算，对应着每个省管辖的所有市、区。其余 5 位为自动生成的编号（包含数字和字母）。下表给出了部分省市的车牌构成状况。

表 2.3 中国车牌构成

省、直辖市	车牌第一位	车牌第二位	举例
北京	京	A B C E F G H J K L M N O P Q Y	京 A56789
上海	沪	A B C D E F G H J K L M N	沪 A56789
天津	津	A B C D E F G H J K L M N	津 A56789
河北	冀	A(石家庄) B C D E F G H J K L M N Q R	冀 A56789
吉林	吉	A (长春) B .....T	吉 A56789
江西	赣	A (南昌) B .....M	赣 A56789
海南	琼	A(海口) B .....E	琼 A56789

下面介绍车牌的规格状况。根据《中华人民共和国机动车号牌》规定的机动车车牌的规格，具体的规格如下图所示。

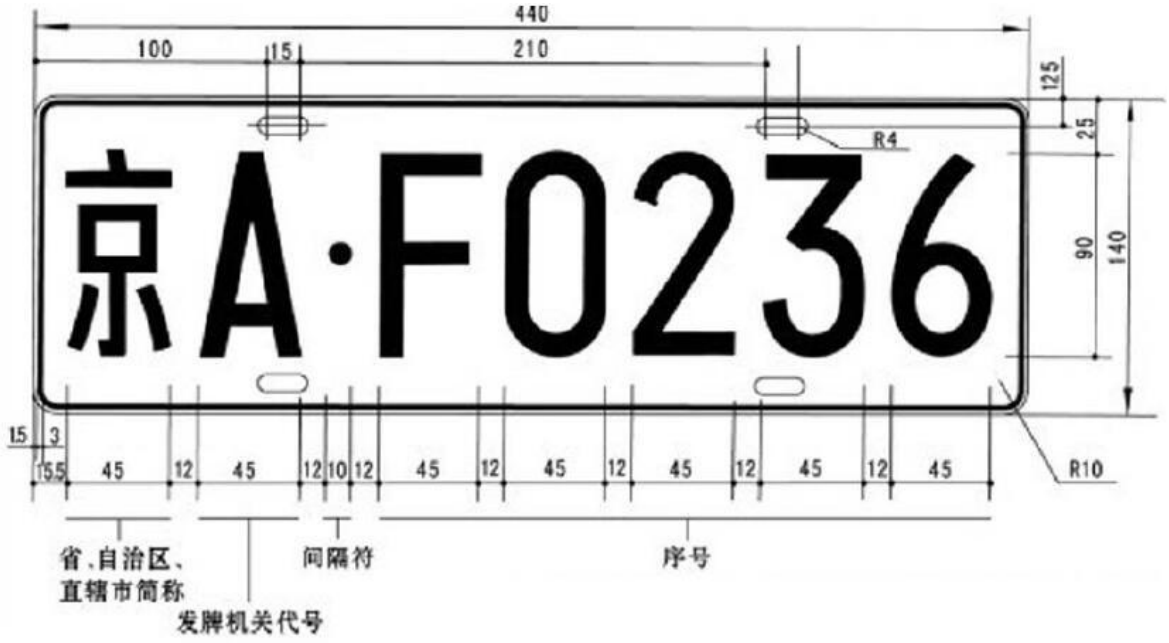

图 2.2 中国车牌规格

2.4 本章小结

本章主要介绍了在进行车牌处理之前那所需要知道的一些预备知识，包括图像的彩色图像的一些基本概念，图像处理时需要的一些概念，以及需要使用的库文件和图形化界面。最后还介绍了中国车牌的主要特点，以方便后续算法的实现。

3 关键算法分析

因为工作流程大致有下面几步：采集原始图像、预处理原始图像、车牌图像的定位、分割字符、识别字符、结果处理。所以主要算法集中在在预处理算法，车牌定位算法字符分割算法以及字符识别算法中。

3.1 图像的预处理算法

在图像识别之前，需要先对牌照进行预处理，以达到尽量消除噪声，抑制无用特性，突出需要识别区域的特征的作用。

3.1.1 灰度化处理

颜色可以分为彩色、黑白、灰度色。灰度化处理可以将彩色图像转化为灰度图。它是使RGB 图像三个分量相等的过程。

灰度化处理的方法主要包含以下几种。

（1）分量法

根据需要，选取 B、G、R 这 3 个分量中的任意一个分量的值，作为灰度图像的灰度值。转换公式如下。

$$f1(i,j)=R(i,j) \quad f2(i,j)=G(i,j) \quad f3(i,j)=B(i,j) \tag{3.1}$$

公式中，i 表示行，j 表示列，f(i,j) 表示在第 i 行，第 j 列的灰度值。

（2）平均值法

计算 B、G、R 这三个分量的灰度值，作为灰度图像的灰度值。转换公式如下。

$$f(i,j)=(R(i,j)+G(i,j)+B(i,j)/3 \tag{3.2}$$

（3）加权平均值法

根据实际生活中，人类眼睛的敏感度要求，给 R、G、B 三个分量乘以不同的权值来求得他们的加权平均值。转换公式如下。

$$f(i,j)=0.3R(i,j)+0.59(i,j)+0.11B(i,j) \tag{3.3}$$

根据式（3.3）中的加权平均的公式，绿色所占比重最大。因此在转换过程中，也可以只用绿色值作为转换后的而灰度。

（4）最大值法

将 R、G、B 三个分量重的最大值找出来，作为灰度图像的值。转换公式如下：

$$f(i,j)=\max(R(i,j),G(i,j),B(i,j)) \tag{3.4}$$

本文采用的灰度值转换方法为加权平均法，该方法可以利用 OpenCV 中自带的 API 函数 cvCvtColor(src, pImageCanny, CV_RGB2GRAY)来实现。其中 src 表示输入图像，pImageCanny 表示输出图像，CV_RGB2GRAY 表示颜色由 RGB 模式转换到 Gray 模式。

3.1.2 灰度分布均衡化

又称直方图分布均衡化。由于图像的亮度可能会不均匀，原始图像中的待测点的颜色会异样。这样可能会导致定位失败，所以我们需要对图像的亮度进行均衡化处理。基本原理是使用累计分布函数变换法，对图像中像素个数较多的灰度级进行展宽操作，对像素个数相对较少的灰度进行缩减操作。函数目的是使直方图平缓、分布均匀、动态范围扩大。

先假定一个图像变换函数。

$$s=T(r)=\int_0^r p_r(\omega)d\omega \tag{3.5}$$

上述公式中，ω是积分变量，$\int_0^r p_r(\omega)d\omega$是累计分布函数。累计分布函数从 0 到 1 单调增加，所以$\int_0^r p_r(\omega)d\omega$可以满足以下两个条件：

（1）在$0\le r\le 1$范围内，$\int_0^r p_r(\omega)d\omega$是关于$T(r)$单值的单调递增的。

（2）在$0\le r\le 1$范围内，$0\le T(r)\le 1$。

再对累计分布函数中的 r 求导。

$$\frac{d_s}{d_r}=p_r(r) \tag{3.6}$$

此外，随机变量η的分布函数已知如下：

$$p_s(s)=p_r(r)=\frac{d}{d_s}[T^{-1}(s)]=[p_r(r)\ \frac{d_r}{d_x}]_{r=T^{-1}(s)} \tag{3.7}$$

将（3.6）代入（3.7）中可以得到：

$$p_s(s)=[p_r(r)\ \frac{d_r}{d_x}]_{r=T^{-1}(s)}=[p_r(r)\ \frac{1}{p_r(r)}]=1 \tag{3.8}$$

上式表明，变化后的函数中，变量 s 在它的定义域内地概率密度函数呈现均匀分布。依据上述条件，用 r 为变量的累积分布函数，它的图像变化函数可以产生灰度级分布均匀的函数。起到了扩展像素分布区间和取值的作用。

对于公式（3.6）的两边进行积分，可以得到下面的公式。

$$s = f(r) = \int_0^r p_r(\mu)d\mu = \frac{1}{A_0}\int_0^r H(\mu)d\mu \tag{3.9}$$

则灰度值均衡化变换公式如下。

$$D_B = f(D_A) = \frac{D_{MAX}}{A_0}\int_0^{D_A} H(\mu)d(\mu) \tag{3.10}$$

其中 D_{MAX} 指最大色阶。对于实际处理中的离散图像，变换公式如下：

$$D_B = f(D_A) = \frac{D_{MAX}}{A_0} = \frac{\mathrm{D_{MAX}}}{\mathrm{A_0}}\sum_{i=0}^{\mathrm{D_A}} H_i \tag{3.11}$$

式中 H_i 表示第 i 级别灰度的像素点个数。

3.1.3 平滑化处理

在工程中，图像的处理、接收、传输和形成的过程中，一定会存在内部干扰和外部干扰。图像平滑或者滤波，就是指消除图像噪声的工作。有两个功能，改善抽出特征和改善图像质量。平滑处理可以在频率域或者空间域进行。空间域常用的是中值滤波、均值滤波和高斯滤波。频率域常用的是维纳滤波、低通滤波、高通滤波。

下面介绍空间域常见的平滑滤波方法。包括以下 3 种。

线性平滑是对用每一个像素点领域的灰度值代替它本身的灰度值。领域大小为 N*N，N 的大小一般为技术。常用的有 3*3 均值滤波器。经过线性平滑滤波后，等价于整个图像经过了一个二维的低通滤波器，在降低噪声的同时也模糊了图像的细节与边缘。

非线性平滑是对线性平滑的一种改进。当像素的灰度值与邻域的灰度值之间的差值大于某一个阙值时，用已知的均值代替，当不大于时，则不改变像素的灰度值。非线性平滑的方法可以在对图像细节影响不大的情况下，消除部分孤立噪声点，但是图像的边缘会有一定失真。

自适应平滑是要根据当地、当时的情况开进行控制的平滑方法。所以这种算法要有一个自适应的目标。根据目的的不同，可以有不同的图像处理方法。因为目标图像与背景图像会有不同的统计特性，即方差和均值不同，为了保留边缘信息，可以使用自适应的局部平滑法。自适应的平滑滤波的目标是尽量不模糊边缘轮廓。比如选择式掩摸平滑法。

高斯滤波器是根据高斯函数来选择权值的的线性平滑滤波器，下式为 0 均值高斯函数：

$$g(x) = e^{-x^2/2\sigma^2} \tag{3.12}$$

公式中高斯滤波器的宽度由 σ 决定。而在图像处理中，需要用到二维 0 均值离散高斯函

数，函数表示如下：

$$g[i,j]=e^{-(i^2-j^2)/2\sigma^2} \tag{3.13}$$

根据高斯函数的可分离性，2 个一维高斯滤波器经过水平方向和竖直方向的卷积，可以得到二维的高斯滤波器。

3.1.4 二值化处理

颜色可以分为彩色、黑白、灰度色。二值化化处理就是由灰度图像转化为只有两种值的图的过程。

二值化处理的公式如下。

$$f(x,y)=\begin{cases}1 & f(x,y)\geq T\\ 0 & \text{其他}\end{cases} \tag{3.14}$$

$f(x,y)$ 表示一幅固定行列的图像上的一个像素值。T 是阙值。

3.1.5 形态学处理

图像的形态学处理包括腐蚀、膨胀、细化、开运算、闭运算等。细化是求一个物体骨架并同时保持它的连通性的过程；闭运算是先膨胀后腐蚀的过程；开运算与之相反。有关图像的形态学处理在此不展开赘述。

3.2 车牌定位算法

车牌定位是车牌识别的首要工作。在原始图像中，一般会包含车牌和车牌的背景图像，在进行下一步的字符分割之前，首先要定位到车牌的位置，并将车牌从原始图像中分离出来。如果车牌定位算法不够准确，字符分割的难度将会增加，甚至无效，从而整个系统的识别率会受到影响。

3.2.1 常用的车牌识别算法

（1）基于形态学法的车牌定位。

在对图像进行二值化处理以后，进行腐蚀膨胀在内的形态学操作。将图像分成若干个小的连通域，做好标记，计算出每个区域的长宽比、面积大小，然后根据预先知道的车牌特征进行区域的筛选。最后确定图像的区域，并实现分割操作。

（2）基于纹理特征的车牌定位。

由于车牌字符信息的纹理相对于它的背景更加丰富，所以可以利用这一特征来定位车牌。首先对图像灰度化，然后根据灰度的变化不同来确定车牌的候选区域。但这种方法对噪声特

别敏感，当背景复杂时，很容易选出非车牌区域，改良方法一般是结合垂直投影算法。

（3）基于颜色空间法的车牌定位。

由于车牌一般具有相对固定的颜色特征，颜色特征的表现很强烈，所以可以统计颜色的特征来实现对车牌的分割。

（4）基于小波变换的车牌定位。

利用平移、伸缩变换对原始图像进行多尺度细化，分解为具有不同频率特征的多个子信号。由于这些子信号有良好的时域特性，所以可以进行局部变换，细分高频和低频。这种方法通常和其他定位方法一起使用。

（5）基于边缘检测的车牌定位。

边缘是图像最基本的特征，当图像中的像素值出现阶跃变化时，就说明出现了边缘。由于图像中两个区域会有比较大的灰度值差异，而同一区域中灰度会均匀的分布，所以可以根据这一特征来进行车牌定位。

边缘检测有几个常用的检测算子：Canny 算子，Prewitt 算子、Roberts 算子以及 Sobel 算子。

（1）Canny 算子

Canny 算子被称为最优化边缘检测算子，它具有相对良好的抗噪性。它的应用比较广泛，可以分为水平和垂直方向。它的计算公式如下。

$$G_i=[f(i,j+1)-f(i,j)+f(i+1,j+1)-f(i+1,j)]/2 \quad (3.15)$$

$$G_j=[f(i,j)-f(i+1,j)+f(i,j+1)-f(i+1,j+1)]/2 \quad (3.16)$$

它的计算模板如下：

$$\begin{bmatrix}-1 & 1\\ -1 & 1\end{bmatrix}\ \begin{bmatrix}1 & 1\\ -1 & -1\end{bmatrix} \quad (3.17)$$

（2）Prewitt 算子

Prewitt 算子是一种 3*3 的梯度算子，它的计算分为两步，先加权平均，然后微分计算。这样产生的优点是有效地抑制噪声，缺点是边缘定位不是非常准确。它有水平和垂直连个分量。它的计算公式如下。

$$Gi=[f(i+1,j-1)+f(i+1,j)+f(i+1,j+1)]-[f(i-1,j+1)+f(i-1,j)+f(i-1,j+1)] \quad (3.18)$$

$$Gj=[f(i-1,j-1)+f(i,j-1)+f(i+1,j-1)]-[f(i-1,j+1)+f(i,j+1)+f(i+1,j+1)] \quad (3.19)$$

它的计算模板如下。

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \tag{3.20}$$

（3）Roberts 算子

Roberts 算子会计算图像对角方向的像素灰度值的差值。缺点是对噪声敏感，但是它定位的边缘比较准确。它的计算公式如下。

$$G(i,j)=\left|f(i,j)-f(i+1,j+1)\right|+\left|f(i+1,j)-f(i,j+1)\right| \tag{3.21}$$

它是个 2*2 的梯度算子，计算模板如下。

$$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{3.22}$$

（4）Sobel 算子

Sobel 算子是一种 3*3 的梯度算子。需要先加权后微分，最后再求梯度。它的计算模板如下。

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{3.23}$$

左边是水平模板，右边是竖直模板，将这个两个量与图像进行卷积，就可以得到水平和竖直方向的领域亮度差分的近似值。假设左边为 Gx,Gx 为横向边缘检测的图像灰度值，右边为 Gy,Gy 为纵向边缘检测的图像灰度值。再假设原始图像为 A，则计算公式如下。

$$Gx=\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathrm{A} \qquad Gy=\begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A \tag{3.24}$$

上述公式展开得到下面的公式。

$$\begin{aligned} Gx &= (-1)*f(x-1,y-1)+0*f(x,y-1)+1*f(x+1,y-1) \\ &+(-2)*f(x-1,y)+0*f(x,y)+2*f(x+1,y) \\ &+(-1)*f(x-1,y+1)+0*f(x,y+1)+1*f(x+1,y+1) \\ &=[f(x+1,y-1)+2*f(x+1,y)+f(x+1,y-1)] \\ &-[f(x-1,y-1)+2*f(x-1,y)+f(x-1,y+1)] \end{aligned} \tag{3.25}$$

$$\begin{aligned} Gy &= 1*f(x-1,y-1)+2*f(x,y-1)+1*f(x+1,y-1)+0*f(x-1,y) \\ &+0*f(x,y)+0*f(x+1,y)+(-1)*f(x-,y+1) \\ &+(-2)*f(x,y+1)+(-1)*f(x+1,y+1) \\ &=[f(x-1,y-1)+2*f(x,y-1)+f(x+1,y-1)] \\ &-[f(x-1,y+1)+2*f(x,y+1)+f(x+1,y+1)] \end{aligned} \tag{3.26}$$

其中 f(x,y)表示原始图像的（x,y）点的灰度值。然后计算梯度的大小，公式如下。

$$|G|=|Gx|+|Gy| \tag{3.27}$$

如果梯度大于某个阈值，则认为（x,y）点为边缘点。它的梯度方向如下。

$$|G|=|Gx|+|Gy| \tag{3.28}$$

实验中发现，Sobel 算子边缘检测得到的信息容易丢失很多边缘，有时会将其他信息作为边缘来处理，所以需要加入其他算法来完善。Canny 算子的公式推导与 Sobel 算子相似。

本课题使用 Canny 算子进行边缘检测。用高斯滤波的方式提高了 Canny 算子的 l 抗干扰能力，用 OpenCV 自带的 cvCanny 函数，用双阈值边缘检测减少假的边缘数。高阈值用于把边缘连接成轮廓，这样具有较少的假边缘，低阈值用于轮廓在轮廓的端点的 8 邻域中找满足的点，根据该点收集新边缘，一直到图像闭合。本文设计了迭代的自适应阈值算法来确定阈值大小。

3.3 字符提取算法

字符提取，就是在已经得到的车牌图像中找到每个字符的图像，进行分割，这样有利于每个字符的的识别。字符提取的基本策略有：基于图像特征的分割策略和基于识别的分割策略。图像特征主要指字符宽度、高度、间距、分布规律等特征，常用方法有垂直投影法和连通域标记法。

3.3.1 基于垂直投影法的车牌定位

将像素纵向上的像素值进行求和，有字符的地方会出现投影较高的情况。而与之相对的，字符中间的间隔区域，一般情况下是不会出现目标像素的，所以投影值应该为 0。考虑到噪声的影响，可以选定一个适当的阈值将字符分割。

3.3.2 基于连通域的车牌定位

众所周知，数字和字母是都是连通的，但是很多汉字不是连通的。所以在车牌字符的分割中，可以先识别数字和字母。根据他们的连通性，很容易定位到每个连通域的起始位置和终止位置。然后根据这连个位置就可以画矩形了。这样就可以得到包含数字或者字母的最小外接矩形。然后根据数字字母与汉字的相对位置，可以确定第一个汉字字符的位置。

3.3.3 基于模板匹配的车牌定位

这个方法先要进行一次垂直投影。然后查找曲线的局部最小值找到波谷。相邻的两个波谷就可以确定一个矩形，把得到的矩形作为模板，从开始位置滑动，找到最佳匹配位置。这种方法很好地解决了汉字不连通的问题，但是程序设计复杂，执行时间长。

3.4 字符识别算法

字符识别在模式识别的范围中，是典型的大类别数识别问题。常用的方法有基于神经网络的识别法，基于模板匹配的识别法和基于特征的识别法。

3.4.1 归一化处理

经过字符分割后的单个字符图像大小并不一定一样，所以为了便于字符识别，需要我们将输入进来的字符图片进行统一尺寸的操作。本文使用线性插值方法实现归一化。使用 OpenCV 自带的 cvResize 函数实现线性插值。

3.4.2 基于模板匹配的车牌识别

模板匹配方法，是一种有效地离散输入模式分类方式。根据图像的形象提取特征，并建立模板库。然后提取图像的特征，与模板库中的所有数据进行匹配。得到最佳结果输出。匹配度是由互相关算子确定的。公式如下。

$$R(i,j)=\frac{\sum_{m-1}^{M}\sum_{n=1}^{N}S^{i,j}(m,n)\times T(m,n)}{\sqrt{\sum_{m-1}^{M}\sum_{n=1}^{N}[S^{i,j}(m,n)]^2}\sqrt{\sum_{m-1}^{M}\sum_{n=1}^{N}[T(m,n)]^2}} \tag{3.29}$$

R(i,j) 为互相关算子，S 为输入字符的图像，$S^{i,j}$ 是输入图像的子图，(i,j) 是输入字符图像在原图中的位子，T(m,n) 是检测模板。计算完成后，用互相关算子最大的模板作为最佳匹配。由于这个过程的图像都是经过二值化处理的，所有公式还可以表示如下。

$$D(i,j)=\sum_{m=1}^{M}\sum_{n=1}^{N}S^{i,j}(m,n)\otimes T(m,n) \tag{3.30}$$

以上是最简单的汉明距离匹配法。较为复杂的是提取特定的特征值进行匹配。模板匹配的缺点是识别率低，受噪声影响大。实际应用中，经常会提取许多个模板进行匹配，处理时间就会显著增强。

本实验中，改进了模板匹配的算法，将图像划分了区域，先由对角线开始，提取对角线上八个连续区域的特征值，然后把图像在水平方向上提取等距离的行特征值，在竖直方向上也提取了三个特征值，总共得到了 14 个特征值进行模板匹配。

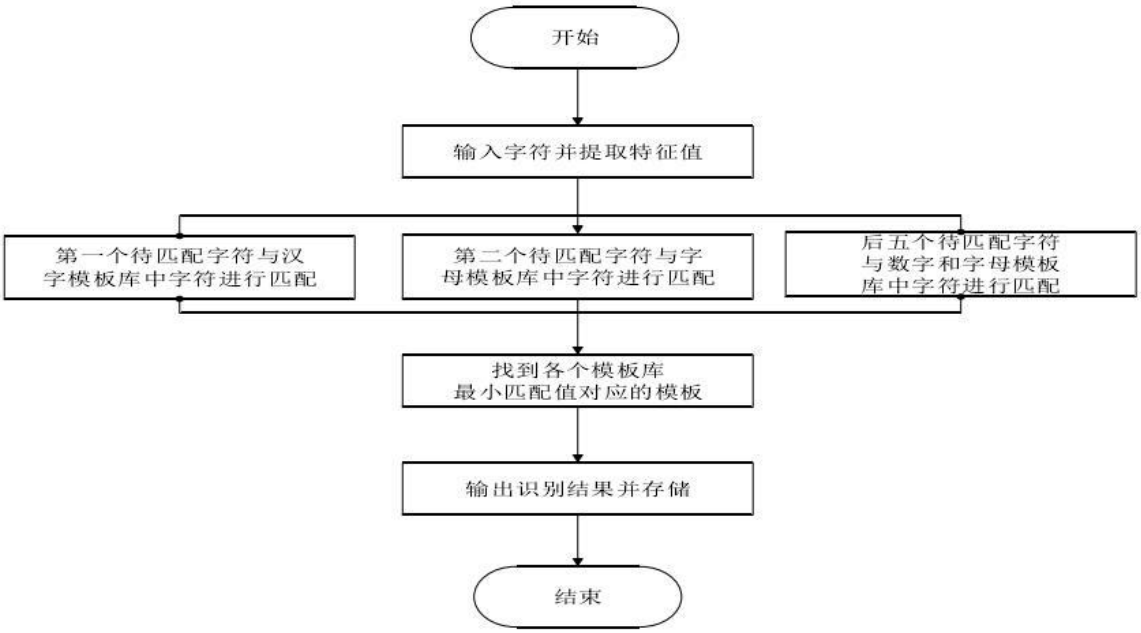

图 3.1 模板匹配算法

3.5 本章小结

本章主要介绍了车牌识别的主要算法，包括了图像的预处理算法，车牌的定位算法，字符分割的算法和字符识别的算法。

4 系统实现

系统实现需要从首先搭建所需的开发平台，即嵌入式平台，然后对所需要的库文件和应用软件进行移植，最后通过使用第 3 章论述的方法编写程序，然后再开发板上仿真。本实验室通过在开发板上移植 Linux 系统，并植入 OpenCV，用摄像头完成对原始图像的采集工作，然后对采集到的车牌图像进行处理和识别工作。然后用 QT 图形化界面实现结果的显示的。

4.1 嵌入式系统概述

嵌入式系统(Embedded System)是后 PC 时代被人们广泛应用的计算平台，它广泛的出现在人们的日常生活中。相对于通用计算机而言，嵌入式系统的硬件和软件资源都受到严格的限制。因为它通常对某些特定的指标，比如体积、功耗、成本、实时性等，有着严格的要求，有时甚至是苛求。

硬件平台是嵌入式系统的基础。嵌入式硬件平台最核心的部件是各种类型的嵌入式微处理器芯片，目前在中国，主要流行过一下几种嵌入式微处理器[3]。

（1）Z80、Intel8080、MC8080 等。

（2）ARM 系列微处理器。

（3）DSP 微处理器。

目前 ARM 系列有从 7 到 11 系列，有 Cortex 系列，还有 XScale 系列。

软件是嵌入式系统的灵魂。目前国内流行的嵌入式操作系统有以下几种。

（1） μC/OS-Ⅱ。

（2）Linux。

（3）Window CE。

（4）VxWorks。

嵌入式系统的应用开发的步骤主要分成以下几步。

（1）启动引导程序（BootLoader）设计。

（2）操作系统的移植（如 Linux 或者 Windows CE），包括根文件系统的设计。

（3）根据应用要求，构建支撑环境，比如嵌入式数据库管理系统的构建、图形界面的构建、嵌入式 Web 服务器的构建等。

（4）应用程序设计，包括操作系统未提供的硬件接口驱动程序设计。

嵌入式系统的主要有以下几种应用领域。

（1）工业控制。

（2）现代农牧业。

（3）智能小区及智能家居。

（4）移动智能终端。

（5）军事领域。

（6）智能交通及汽车电子。

（7）机器人。

4.2 总体设计方案

本车牌识别系统的设计，主要分为以下四个部分。

（1）在嵌入式 Linux 操作系统中移植 OpenCV 函数库文件，并编程和仿真。

（2）嵌入式与 Qt 结合，显示实验结果。

（3）交叉编译执行文件。

（4）以 ARM9 开发板为终端目标平台，移植程序。具体流程如下。

图 4.1 移植程序流程

4.2.1 搭建开发环境

开发环境，是指在基本硬件基础上，支持系统开发设计的一组工具的集合。本系统开发中，要使用的软件有：OpenCV、Qt Creator、arm-linux-gcc 编译器、gcc 编译器。他们各自的用途如下表。

表 4.1 主要软件用途

名称	用途
Gcc 编译器	Linux 系统下的多平台编译器。本课题中辅助 OpenCV 函数库编译出车牌图像。
Arm-linux-gcc	Linux 系统下的交叉编译器。本课题中用来移植 OpenCV 软件和 Qt,使 Gcc 生成的文件能再 ARM 上执行。
Qt Creator	Linux 系统下的 Qt 图形化界面处理软件，可生成应用程序
OpenCV 函数库	计算机视觉库，本课题使用用 Linux 版本。

然后安装 Linux 的 Ubuntun 系统，如下图所示：

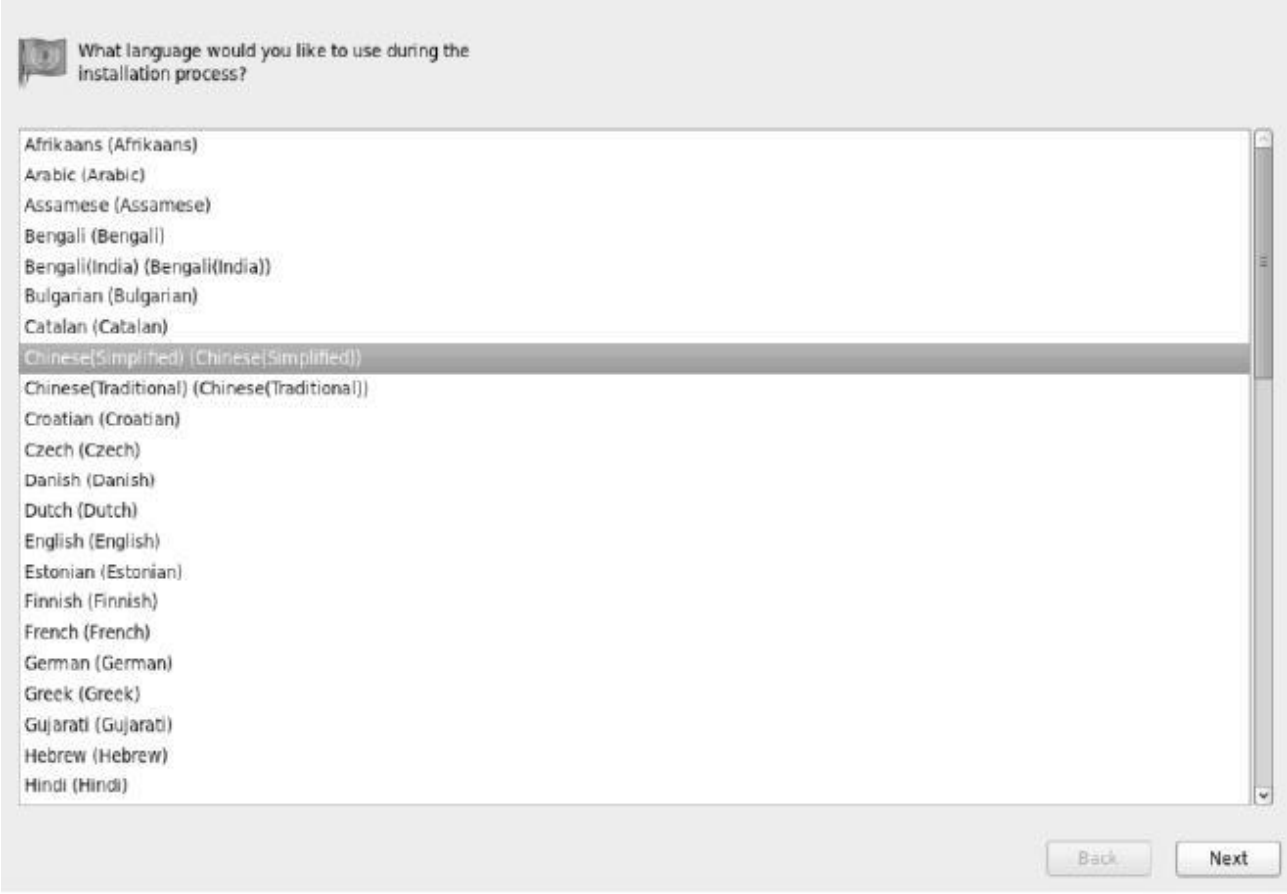

图 4.2 Linux 安装

4.2.2 安装交叉编译工具

交叉编译工具，就是指当硬件平台和主机的系统不同时，通过交叉编译器可以在宿主机上编辑，在目标机器上编译结果。因为嵌入式 Linux 不支持 OpenCV 自带的 Highgui(动态链接库文件)，所以 OpenCV 是处理后的车牌图像，在嵌入式目标平台上没有办法直接显示。这就需要用到我们在上文提到的 QTE 图像界面了。移植 QT 库就需要用到 arm-linux-gcc 编译器。

4.2.3 交叉编译 OpenCV

OpenCV 也是一个需要移植的函数库。本设计使用 OpenCV2.2 版本。只需要将 OpenCV 的安装包复制粘贴到指定的文件夹下，并在 CMake 中设定相关的路径就可以了。之后会出现这样的界面，按下图的操作执行。

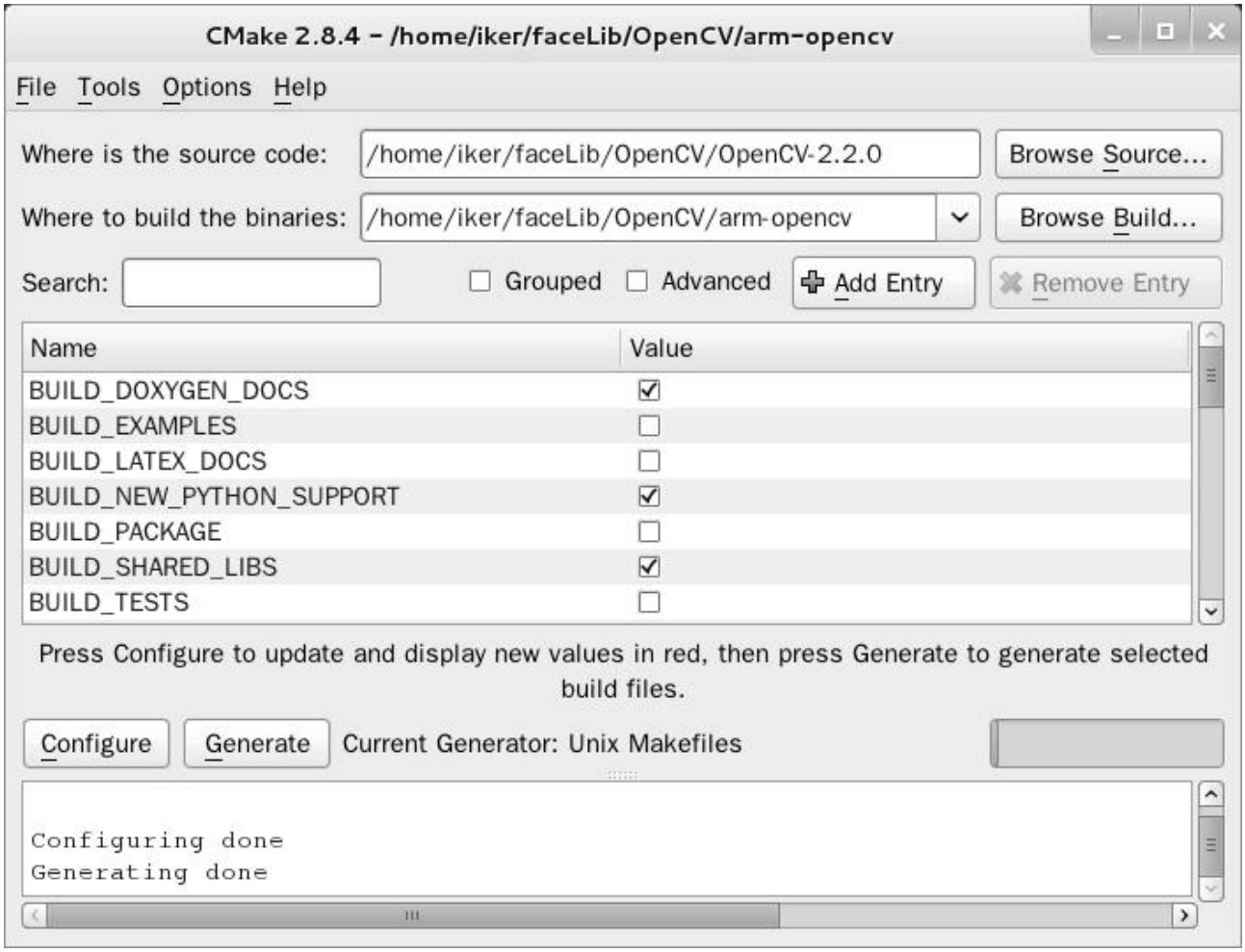

图 4.3 OpenCV 安装

4.2.4 整合 QtCreator

QT 是完全面相对象的，拥有丰富的 API，有大量的开发文档，支持 XML，集成了 Webkit 引擎从而实现本地与 Web 内容的无缝集成。QT/Embedded 面相嵌入式的 QT 版本。QT 安装如下图所示。

图 4.4 整合 Qtcreator

4.3 Linux 系统与动态库的移植

上面介绍了在个人计算机上移植嵌入式操作系统，QT 库和 OpenCV 库，接下来我们在 ARM 平台上移植这些配置。OpenCV 与 QT 的移植与主机上的类似。嵌入式的移植包括了三个部分，文件系统、内核和 u-boot。

u-boot 是启动引导程序的一种。作用是实现初始化看门狗定时器、初始化堆栈、关中断等初始化操作。为了保护硬件系统，就需要内核来限制应用程序访问。文件系统是管理用户文件的内核层。进行完以上交叉编译之后，就可以将其烧写到 ARM 平台的存储器中了。

4.4 相关函数实现

根据图 2.1，相关的实现步骤主要有以下几步。

（1）使用从图片库中载入原始车牌图片。使用函数为 OpenCV 自带的 cvLoadImage()函数。

（2）使用加权平均法进行灰度值转换。使用函数为 OpenCV 自带的 cvCvtColor()函数。

（3）使用高斯平滑滤波得到滤波后的图片。使用函数为 OpenCV 自带的 cvSmooth()函数。第三个参数设置为 CV_GAUSSIAN。

（4）使用 Canny 边缘检测算法检测车牌图像的边缘，输出二值化图像。调用 OpenCV 自带的 cvCanny 进行双阈值检测。其中第一个阈值使用迭代法实现自适应阈值函数 AdaptiveThreshold()来确定。

（5）使用函数 PlateAreaSearch()实现车牌定位。

首先，对于图像进行细化，把图像掏空，只留下最边缘的数值，并在细化后对每一行像素值求和得到每行的数据量。

然后，以 20 行为一组进行扫描，得到数据量最大的 20 行为车牌的区域，由此确定车牌上下边界 plate_n 与 plate_s。

然后，用一个矩形块从左向右滑动，圈出数据量最大的就是车牌。

最后，将车牌线性插值，归一化为高度为 40 像素的车牌图像。重新进行（2）（4）步。

（6）使用垂直投影的办法进行字符分割，函数为 SegmentPlate()。首先获得第二个字符的开始位置和结束位置，然后根据第二个字符的位置，确定第一个字符的区域，同时确定了后面 5 个字符的区域。

（7）使用模板匹配的办法进行字符识别。

用函数 OnShibiecar 将现有字符分为三组，第一个字符为汉字字符，第二个为字母字符，后面五个为一组包括字母字符和数字字符。

用函数 CodeRecognize()进行模板匹配。模板中一共以 14 个特征值，先由对角线开始，提取对角线上八个连续 10*10 矩形区域的特征值，然后把图像在水平方向上提取等距离的行特征值，在竖直方向上也提取了三个特征值，总共得到了 14 个特征值进行模板匹配。对于前八个值，与模板相差小于 2 即配对成功一次。第 9 个值与模板相差不大于 8 即为配对成功一次。后面几个特征值大于 5 时，要求与模板相差不大于 1，后面几个特征值大于 5 时，必须与模板值相等才可以算配对成功一次。最后保留配对成功数最多的模板对应字符作为匹配结果。

以上步骤的函数表如下。

表 4.2 车牌识别主要函数表

函数名	返回值类型	函数参数	作用描述
cvLoadImage	IplImage*	const char* filename, int flags=CV_LOAD_IMAGE_COLOR	载入图片
cvCvtColor	void	const CvArr* src, CvArr* dst, int code	转化为灰度图
cvSmooth	void	const CvArr* src, CvArr* dst, int smoothtype=CV_GAUSSIAN, int param1=3, int param2=0	高斯平滑滤波
Threshold	void	IplImage*Image, IplImage*Image_O	边缘检测
cvCanny	void	const CvArr* image,CvArr* edges,double threshold1,double threshold2, int aperture_size=3	边缘检测被 Threshold 调用
AdaptiveThreshold	int	int t, IplImage *Image	迭代法确定 cvCanny 第一个阙值
PlateAreaSearch	int	IplImage *pImg_Image	查找车牌区域
cvResize	void	const CvArr* src, CvArr* dst, int interpolation=CV_INTER_LINEAR	车牌归一化
SegmentPlate	int	void	分割车牌
CodeRecognize	int	IplImage *imgTest, int num, int char_num	模板匹配

4.5 相关 UI 设计

根据上文第三章的主要流程，该车牌识别系统主要 UI 设计如下。

1 需要一个按钮。目的是为了打开摄像头获取原始图片；

2 需要一个按钮。用于从图片库中载入图片到文本框中显示；

3 需要一个文本框。用于显示需要识别的图片；

4 需要一个按钮。目的是为了开始运行车牌识别算法；

5 需要一个按钮。用于退出界面；

6 需要一个文本框。用于显示识别出来的车牌号码；

根据以上需求建立 UI，如下图。

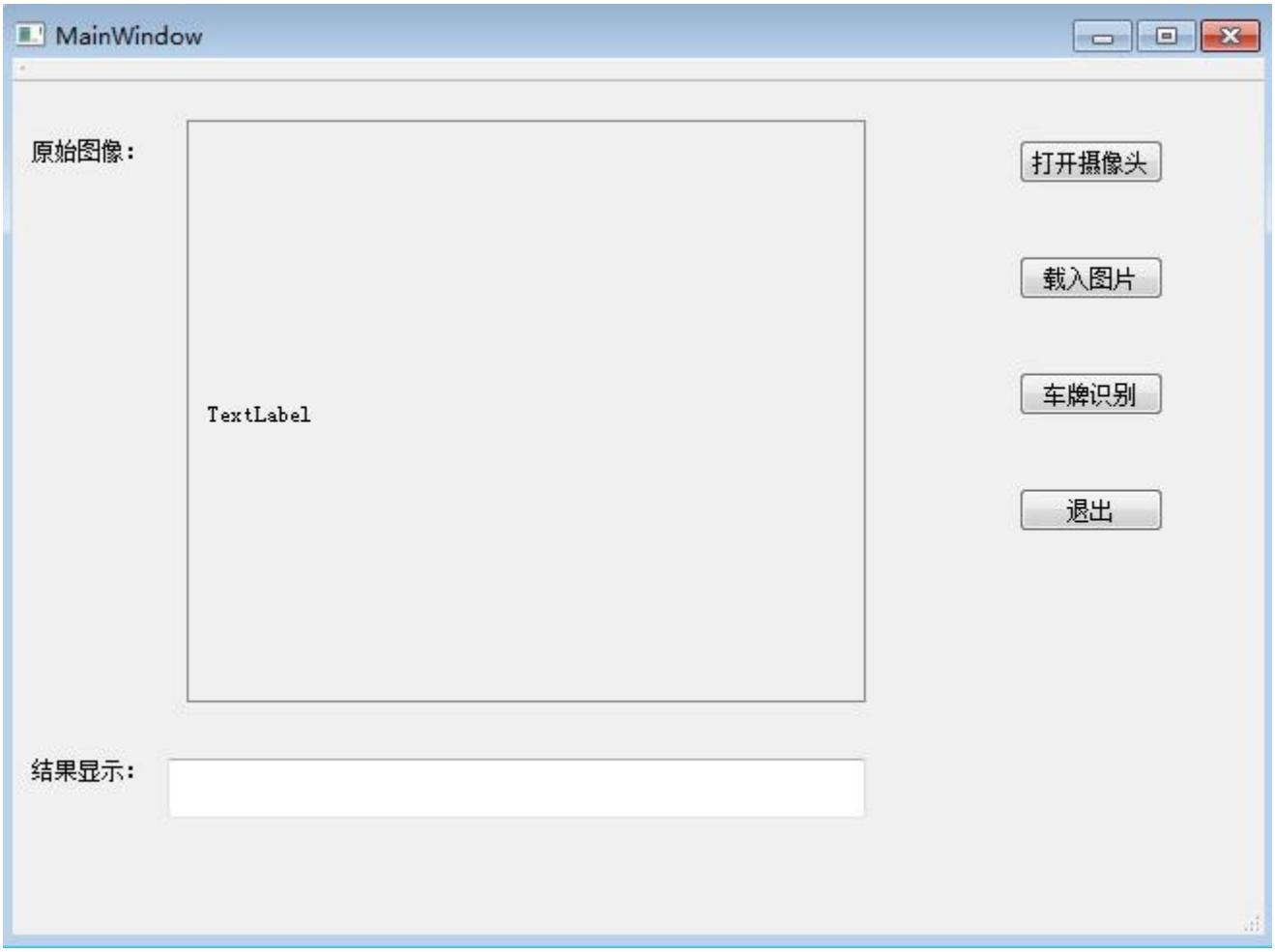

图 4.5 车牌识别系统 UI

4.6 识别程序的嵌入式实现

（1）下载车牌识别程序到 AMR 编辑平台。

（2）通过设定 Qt Creator 的 ARM 版 QMake 处理之后的文件，作为这项根目录下的以"-build-desktop"结尾的文件。将这个文件拷贝进入 SD 卡，然后用 ARM 挂载。在 SD 卡目录

下输入一下命令。

#chmod 777 QtOpenCV　//修改文件权限

#./QtOpenCV -qws

（3）这个命令可以设定车牌识别系统的显示方式是主窗口模式。-qws 让 QtOpenCV 成为识别系统的服务器。

（4）完成上述设定后，还需要完成最后的环境配置。包括指定一些库文件的路径等。然后就可以运行系统了。

（5）实验的结果 1 如下图所示。

实验结果 1 中，车牌被完美的识别了出来。

图 4.6 实验结果 1

图 4.7 实验结果 1 灰度化图

从实验结果 1 的灰度化图可以看出，灰度化后车牌清晰可见，不影响识别效率。

图 4.8 实验结果 1 边缘检测图

实验结果 1 的边缘检测图可以看出，轮廓被很好地提取了出来，图像中的数据量与灰度图相比大大减少了，提高了定位算法的效率。

图 4.9 实验结果 1 车牌定位图

实验结果 1 的车牌定位图可以看出，车牌被很好地定位了出来，上下左右边缘清晰可见。

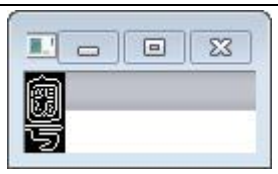

图 4.10 实验结果 1 字符 1 图

实验结果 1 的字符 1 分割可以看出，字符“粤”被很好地分割了出来，上下左右边缘清晰可见。

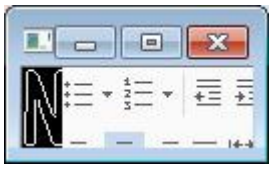

图 4.11 实验结果 1 字符 2 图

实验结果 1 的字符 2 分割可以看出，字符“N”被较好地分割了出来，上下左边缘清晰可见，右边缘缺少了一条轮廓线，但并没有影响识别效果。

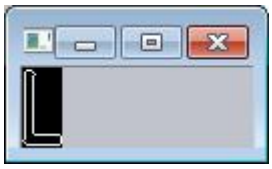

图 4.12 实验结果 1 字符 3 图

实验结果 1 的字符 3 分割可以看出，字母“L”被很好地分割了出来，上下左右边缘清晰可见。

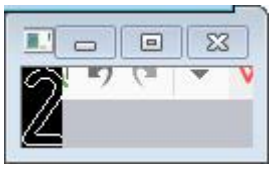

图 4.13 实验结果 1 字符 4 图

实验结果 1 的字符 4 分割可以看出，数字“2”被很好地分割了出来，上下左右边缘清晰可见。

图 4.14 实验结果 1 字符 5 图

实验结果 1 的字符 5 分割可以看出，数字“2”被很好地分割了出来，上下左右边缘清晰可见。

图 4.15 实验结果 1 字符 6 图

实验结果 1 的字符 6 分割可以看出，数字“8”被很好地分割了出来，上下左右边缘清晰可见。

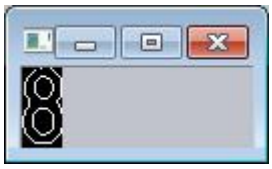

图 4.16 实验结果 1 字符 7 图

实验结果 1 的字符 7 分割可以看出，数字“8”被很好地分割了出来，上下左右边缘清晰可见。

（6）实验的结果 2 如下图所示。

实验结果 2 中，发现第二个字符识别有误，主要是因为 X 与 K 字符有半部分的图像非常相像，在截取图像时，由于图像都被细化过，特征值很相像，导致了识别率有一定的降低，但是总的来说，大部分字符的识别率还是相对较高的。

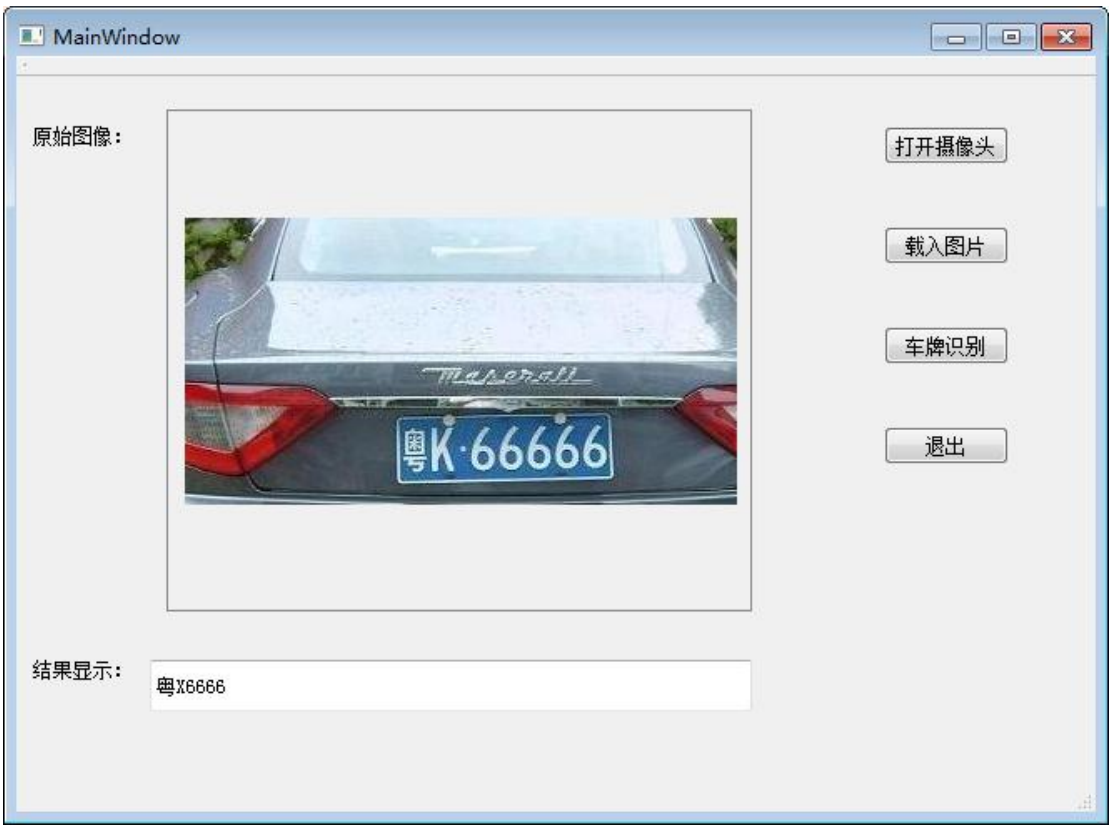

图 4.17 实验结果 2

图 4.18 实验结果 2 灰度化图

从实验结果 2 的灰度化图可以看出，灰度化后车牌清晰可见，不影响识别效率。

图 4.19 实验结果 2 二值化图

实验结果 2 的边缘检测图可以看出，轮廓被很好地提取了出来，图像中的数据量与灰度图相比大大减少了，提高了定位算法的效率。

图 4.20 实验结果 2 车牌定位图

实验结果 2 的车牌定位图可以看出，车牌被很好地定位了出来，上下左右边缘清晰可见。

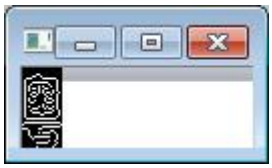

图 4.21 实验结果 2 字符 1 图

实验结果 2 的字符 1 分割可以看出，字符“粤”被很好地分割了出来，上下左右边缘清晰可见。

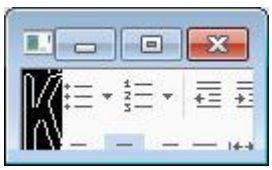

图 4.22 实验结果 2 字符 2 图

实验结果 1 的字符 2 分割可以看出，字符“K”被较好地分割了出来，上左右边缘清晰可见，下边缘缺少了一条轮廓线，实际影响道了识别效果，字符被识别为了 X。

图 4.23 实验结果 2 字符 3 图

实验结果 2 的字符 3 分割可以看出，数字“6”被很好地分割了出来，上下左右边缘清晰可见。

图 4.24 实验结果 2 字符 4 图

实验结果 2 的字符 4 分割可以看出，数字“6”被很好地分割了出来，上下左右边缘清晰可见。

图 4.25 实验结果 2 字符 5 图

实验结果 2 的字符 5 分割可以看出，数字“6”被很好地分割了出来，上下左右边缘清晰可见。

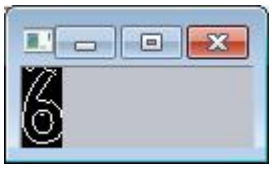

图 4.26 实验结果 2 字符 6 图

实验结果 2 的字符 6 分割可以看出，数字“6”被很好地分割了出来，上下左右边缘清晰可见。

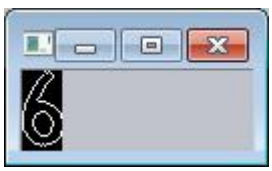

图 4.27 实验结果 2 字符 7 图

实验结果 2 的字符 6 分割可以看出，数字“6”被很好地分割了出来，上下左右边缘清晰可见。

4.7 本章小结

本章主要介绍了嵌入式平台的基本知识，系统的实现，包括搭建开发环境，安装交叉编译工具，整合 QtCreator, 裁剪并移植 Linux 系统，移植识别系统。最后给出了实验结果，已经主要函数和系统 UI。

结 论

车牌识别系统，主要功能是使用摄像设备采集原始车辆图像，然后通过采用相应的人工智能、机器视觉、图像处理等技术，在原始图像中检测出车牌的位置并提取车牌。随后分割车牌字符图像，然后用字符识别技术识别出车牌中的字母、数字以及各种国家的字符，最后得到车牌号码。

本文首先阐述了了课题研究的背景。通过分析，说明了实现一套便携的车牌识别系统的意义。

其次交代了在进行车牌识别之前所需要了解的相关背景知识。

第三章主要分析了车牌识别过程中的主要算法。其中包括图像的预处理算法（灰度化、直方图均衡化、平滑化、二值化、形态学处理等）。还介绍了常见的车牌定位算法（形态学定位法、纹理特征定位法、颜色空间法、小波变换法、边缘检测法），详细介绍了边缘检测的实现。在简单介绍了字符提取算法后，我们详细介绍了基于模板匹配匹配的字符识别算法。

在第四章，我们主要介绍了系统移植的过程，包括对于宿主机器的 Linux 安装，完成交叉编译环境的建立，移植内核和启动引导程序，完成 Qt 开发环境的创建与移植，完成 Opencv 的移植与编译安装，并展示了实验结果。

本设计只是针对静止的图像进行分析，但是在实际生活中，往往需要动态地捕捉视频文件中的车牌，对此需要编写更为复杂的程序编写。

本论文提出的字符分割算法只适用于单排的车牌识别，事实上生活中有一些车牌是双排显示的，这样的车牌采用垂直直方图来进行字符分割的方法显然是不可取的，需要研究出其他的算法。

本文对字符识别的特征值得提取方案还可以进一步优化，部分相似的字母比如 K 与 X 识别准确率有待提高。

致 谢

到此，四年的充实而紧张的大学生活就算圆满结束了。非常感谢我的毕业设计导师孙晋给我在最后这一段时间的悉心指导。孙晋老师是我见过的最亲切的导师，在论文攻坚克难的时期，是孙晋老师不断鼓励我，督促我，不厌其烦的为我解释各种算法的实现过程，尤其是在最后的字符识别的过程中，我们遇到了车牌识别系统中最难识别的部分，在我一筹莫展的时候是孙晋老师再次给予我援手，让我完成了在我眼中几乎不可能完成的任务。大学四年的生活中，孙晋老师经常与我谈心，除了学习以外，我们还会聊他的出国经历，由于我研究生要出国读书，孙晋老师给我提供的经验让我受益匪浅。与一般导师不同的是，孙晋老师还会和我聊聊工作上的故事，我有感情上的困惑也会与孙晋老师沟通，除了师生关系外，我们发展出了深厚的友谊，这在我的求学生涯中是第一个，我非常珍视这段难能可贵的师生情与朋友情，我想我们会做一辈子的好朋友。

除了孙晋老师以外，我还要感谢我的前舍友诺穆奇同学，是他为我分享了他自己买的最新版的 OpenCV 的资料，让我在查询相关资料时候得心应手，高效地完整了论文的书写。我还要感谢我的现舍友宋佰超同学，是他在我忙于写论文之际，关心我的身体健康，拉我出去锻炼身体，帮我按摩酸痛的肩膀，保证了我的身体健康，提高我的写代码的效率。我还要感谢带我进入嵌入式殿堂的符意德老师，是他送我了一本他自己编写的最新版的嵌入式系统原理的书籍，让我领略到了最前沿的嵌入式知识。我还有感谢我最好的朋友周丹丹同学，在我心情沉重的时候，她同学会拉我出去散步，陪我聊天看电影，让我重新振作起来，继续投入到高效的工作中去。我还要感谢我跟我同一个教研室的冒晶晶同学，她做的是基于嵌入式的人脸识别，是她主动帮我解答关于驱动程序的相关问题。

再次感谢所有帮助过我的各位尊敬的老师和乐于助人的同学们，你们的帮助我将铭记于心。

参 考 文 献

[1] 吴秀丽.车牌识别系统的设计与实现[D]: 硕士学位论文.北京: 北京交通大学, 2006.
[2] Kerry L.Clines.Intelligent Transportation Systems[J].Better Road, 2005, 75(7): 75-76.
[3] 符意德，徐江. 嵌入式系统原理和接口技术(第 2 版)[M]. 清华大学出版社, 2007.
[4] 百度百科 网址：http://baike.baidu.com/view/86282.htm?fr=aladdin.
[5] 工良红,冷建华. 汽车倾斜牌照中字符的定位于提取[J]. 电讯技术.2003.4:59-62.
[6] R.Mullot, C. Oliver, J.L. Bourdon. Automatic extraction methods of container Identity number and registration Plate so fears, International Conference on industrial Electronics, Control and instrumentation, 1991, 1793-1744.
[7] Thanongsak, Sirthinaphong. Extraction of car license Plate using motor vehicle Regulation and character Pattern recognition[J], IEEE Trans. Electronic Computers, 1998, 559-563.
[8] S H Parks, K.I. Kim, K Jung,H-J Kim. Locating Car Licence Plate Using Neural Networks[J], Electronics Letters, 2007,(17)3:1475-1477.
[9] 红梅. 基于多帧字符识别的车牌识别系统的研究[D].南京理工大学硕士学位论文，2012.
[10] Zhu Wei Gang, Hou Guo Jiang, Jia Xing. A Study of Location Vehicle License Plate Base on Color Feature and Mathematical Moprhology[C].In:Proceeding Of 6th International Conference on Signal Processing(ICSP,02),Beijing,2002,748-751.
[11] 赵雪春，戚飞虎. 基于彩色分割的车牌自动识别技术[J].上海交通大学学报，1998,32(10):4-9.
[12] 张引，潘云鹤.彩色汽车图像牌照定位新方法[J]. 中国图图形学报，2001, 6(4):374-377.
[13] 张玲，刘勇，何伟. 自适应遗传算法在车牌定位中的应用[J].计算机应用, 2008 28(1):184-186.
[14] 王玫，王国宏.利用伴生与互补颜色特征的车牌定位新方法[J].计算机工程与应用, 2008, 23(7):256-259.
[15] 崔思远. 一种基于区域综合特征的车牌定位算法[D].西安：西安电子科技大学.2007:4-9.
[16] Gloria Bueno Garcia,Oscar Deniz Suarez,Jose Luis Espinosa Aranda, et al. Learning Image Processing with Opencv[J].Packt Publishing - ebooks Account,2015.