XXXXXX

毕业设计说明书（论文）

作　　者：叶超　　学　号：XXXXXXXXXX

学院(系)：电子电气工程系

专　　业：电子科学与技术

题　　目：基于模式识别与神经网络的车牌号码识别系统设计

指导者：温 XX　　讲师

评阅者：XX　　副教授

2014　年 6　月

毕 业 设 计 说 明 书 （论 文） 中 文 摘 要

本设计主要是研究了目前两大主流字符识别方案，模式识别与 BP 神经网络识别方案。以 MATLAB 软件为平台设计系统， 比较了两种字符识别方案结果的准确率，最终得出结论。本系统主要包括图像采集、图像预处理、车牌定位、字符分割、字符识别五个部分组成。本文的图像预处理是将图像经过灰度化，图像增强、边缘提取二值化等操作转化成便于定位的二值化图像。图像定位是采用基于边缘检测、形态学处理的算法，该算法定位精度高。字符分割采用的方法是在二值化后的车牌部分寻找连续有文字的块，大于设定阈值的长度部分则切割。最终本文将通过比较模式识别与 BP 神经网络识别的算法，通过结果分析得出优点与不足，并为以后的进一步研究与改进提出了意见和建议。

关键词 模式识别 BP 神经网络识别 MATLAB

毕 业 设 计 说 明 书 （论 文） 外 文 摘 要

Title Based on pattern recognition and neural network license plate number recognition system design

Abstract

This design aims at studying the two current main character recognition program, pattern recognition and BP neural network recognition program.Based on MATLAB software platform to design systems, this design compares two character recognition program results' accuracy, finally it reaches the conclusion.The system mainly includes five parts: image acquisition, image preprocessing, license plate location, character segmentation and character recognition. Image pre-processing of the paper is the image through the gray, image enhancement, edge extraction binarization operation into easy positioning of the binary image.Using the character segmentation method is part of the license plate binarization looking after consecutive blocks with text, setting the length of the part is greater than cutting the threshold .This article will compare the final pattern recognition and BP neural network algorithm, the analysis of the results obtained by the advantages and disadvantages.For future further research and improvement put forward opinions and suggestions.

Keywords Pattern recognition BP neural network recognition MATLAB

目　　录

1　绪论

近年来，随着经济的发展，人们生活水平的提高，私家车数量迅速增加。城市大力发展交通设施已跟不上车辆增长的速度，　即便是大力发展公共交通设施也难解决现已存在的交通拥挤情况。　由于城市空间的严格限制以及政策资金的支持有限，　使道路基础设施的建设受到了严格限制。　因此要保持交通建设与现代化管理的齐头并进，　建设交通基础设施的同时，大力发展现代智能交通系统（　Intelligent Transport System,简称 ITS）[1]。成为急需解决的问题。

1.1　课题背景及意义

车牌号码识别系统是智能交通系统的核心部分，汽车的号牌是车辆身份的象征，类似于商品的条形码。车牌号码识别技术在交通系统中占据及其重要的位置　[2]。该技术应用范围也相当广泛，如：公共场所出入口车辆管理、交通流量监测、车辆违章记录、车辆被盗追回等等，　使用价值不言而喻。对促进交通稳固持续发展有着极大地推进作用。

1.1.1　车牌识别系统国内外研究现状

自 1998 年起，鉴于车辆普及的必然趋势，外国科学家就已经开始了对车牌识别系统的研究 [3]。当时的主要研究途径是分析车辆的图像，自动提取车牌信息从而确定汽车牌号。在研究过程中，似然运用了很多的技术，但外界环境的多种多样，如外界光线变化、 光路中的灰尘、 季节导致的环境变化及车牌本身的不清晰。　使得车牌识别系统研究一直得不到较好的研究成果。　研究过程中采用了大量的数值运算，　并没有考虑到针对具体情况分办法解决。　当时国外的研究机构为了解决图像的恶化采取的办法是用主动红外照明摄像或使用特殊的传感器来提高图像质量。　但这么做造成的大量投资成本使应用领域变得相对狭小，也打不成大范围推广使用的目的。当时以色列 Hi-Tech 公司的 See/Car System系列，香港 Asia VisionTechnology公司的 VECON 产品以及新加坡 Optasia 公司的 VLPRS 系列都是相对成熟的产品。但　VECON 产品和 VLPRS 产品主要适用于香港和新加坡的车牌，　Hi-Tech 公司的 See/CarSystem通过多种变形的产品来分别适应各个国家的车牌。只有　See/Car Chinese 系统可以对中国大陆的车牌进行识别，　但当时这些产品都存在极大缺陷，　且对车牌中汉字无法识别。　这些国家的产品虽然不同，　但几乎都采用的基于车辆探测器的系统，　导致了设备投资巨大。国内在 90 年代也开始了对车牌识别系统的研究，目前相对成熟的产品有中科院

自动化研究所汉王公司的“ 汉王眼 ” ，另外亚洲视觉科技有限公司、深圳吉通电子有限公司、中国信息产业部下属的中智交通电子有限公司等都有自主研发的产品，另外全国多家研究室也都做过类似研究，为了提高识别率，通常处理都采用了一些硬件探测器和其他辅助设备。

国内在 20 世纪 90 年代进行的相关研究比较好的有：上海交通大学戚飞虎提出了基于彩色分割的牌照识别方法 [4]；华中科技大学黄心汗提出了基于模板匹配和神经网络的牌照识别方法；西安交通大学的郑南宁等人提出了多层次纹理分析的牌照识别方法。车牌识别技术的研究开发出了一些适合在我国使用的识别车辆牌照的产品如：亚洲视觉生产的 VECON-VIS 车牌号码识别系统、成都西图科技有限公司生产的 CIAS-T2003 车牌识别稽查系统、上海高德威公司的车牌号码识别系统以及北京汉王公司的嵌入式一体化车牌辨识仪等，这些产品牌照识别率都达到了 95%以上。但由于车牌号码识别受外界环境影响较大，所生产产品的成本居高不下，无法大规模的推广使用，因此在车牌号码定位和识别的算法改进方面还需要做大量研究。

1.1.2 车牌号码识别技术难点

目前车牌识别率较低的主要原因有以下几种：

（1）我国车牌自身特征的复杂性 [5]。

（2）汉字、字母、数字组合，汉字的复杂程度远大于数字和字母。

（3）我国车牌颜色种类多。大致分为黄底黑字，蓝底白字，白底黑字和黑底白字
四种。

（4）一些人为因素，如道路交通环境、车牌污损、或车牌偏斜角度过大。

（5）我国车牌格式多，包括：民用车牌、公交车牌、公安机关车牌、军用车车牌、
外交车牌、特种车牌、消防车牌等。

（6）车牌悬挂方式多样化，由于不同汽车公司出产的汽车型号和外形各不相同，致使了车牌的悬挂位置不唯一。

（7）外界环境干扰。光照有强有弱，照射角度不同，不同气候，车牌反光程度都
直接关系到所采集图像的亮度特征，从而影响识别率。

（8）外界背景的多样性 [6]。如背景中出现和车牌特征相似的长方形区域，这些会

干扰车牌定位，造成区域的误判。

（9）拍摄角度问题，如拍摄车牌的角度越小，车牌在图像中的形变越小，识别准

确率越高。

1.2 课题设计任务

本次设计将结合国内外车牌识别系统研究现状，拟设计一个由图像输入到系统处理得到车牌字符输出的车牌识别系统。该系统将重点放在识别系统中各步骤的优化上。采用并分析现今应用广泛的两种字符识别方法：模式识别和神经网络字符识别。并且制作出一个可视化的界面，便于操作使用，以及结果呈现。

1.3 本文的章节安排

第一章节中介绍了车牌识别系统的背景和其开发的重要意义，以及国内外研究的现状。

第二章节中将对拟采用的设计方案和用到的 MATLAB 软件编程环境进行可行性论证。

第三章节中详细介绍了车牌号码识别系统的设计过程，并对模式识别的结果进行了整理分析。

第四章节中介绍了 BP 神经网络字符识别的算法理念，同时介绍了我所设计的 BP 神经网络字符识别的网络结构，训练过程，并对最终识别的结果进行了分析。

第五章节呈现了车牌识别系统设计的可视化操作界面。本文的最后对这次的毕业设计进行了总结和致谢。

2　系统方案论证

通过搜集资料，以及学习他人实验经验，本次设计主要任务是采用 MATLAB 软件编程来实现车牌号码的识别。 下面将分别对软件开发的可行性和系统的设计方案进行论证。

2.1　软件开发可行性论证

MATLAB 是由美国 mathworks 公司发布的主要面对科学计算、可视化以及交互式程序设计的高科技计算环境。 MATLAB 的独特之处在于它将数值分析、矩阵计算、科学数据可视化和非线性动态系统的建模和仿真等诸多强大功能集成在一个易于操作调用的可视化窗口中 [7]，这为科学研究、工程设计等需要进行有效数值计算的众多科学领域提供了一种全面便利的解决方案。代表了当今计算机科学的先进水平。

也有人采用 C 语言来编写车牌识别程序， 但 MATLAB 的优点在于摆脱了传统非交互式程序设计语言（如 C、Fortran）的编辑模式。如同 FORTRAN 和 C 等高级计算机语言使人们摆脱了需要直接对计算机硬件进行操作一样， 有着新时代计算机语言之称的 MATLAB ，利用丰富的已有函数资源，使编程人员不再面对繁琐的程序代码，突出了简洁的编程理念， MATLAB 采用更直观更符合人们操作和思维习惯的代码语句。代替了 C 和 FORTRAN 语言的冗长代码。提供给用户最直观最简洁的程序开发环境。采用 MATLAB 作为本设计的开发环境主要基于其一下几点优点：

（1）语言简洁，使用方便，函数库及其丰富。

（2）运算符丰富，由于 MATLAB 是由 C 语言编写而成的， MATLAB 包含了所有 C 语言的运算符，为程序编写的简短奠定了基础。

（3）没有严格的语法限制，程序设计相对自由

（4）程序兼容性好，基本上不做修改就可运行于大多数计算机操作系统上。

（5）MATLAB 图形功能强大，在图形绘制，数据的可视化上远优于 FORTRAN 和 C 语言，同时 MATLAB 还具有很强的编辑图形界面能力 [8]。

2.2　系统设计方案论证

车牌号码识别是通过采集含有车牌的照片， 通过处理提取出照片中的车牌并识别出计算机所能识别的车牌号码信息加以存储。主要设计流程如图 2.1 所示。

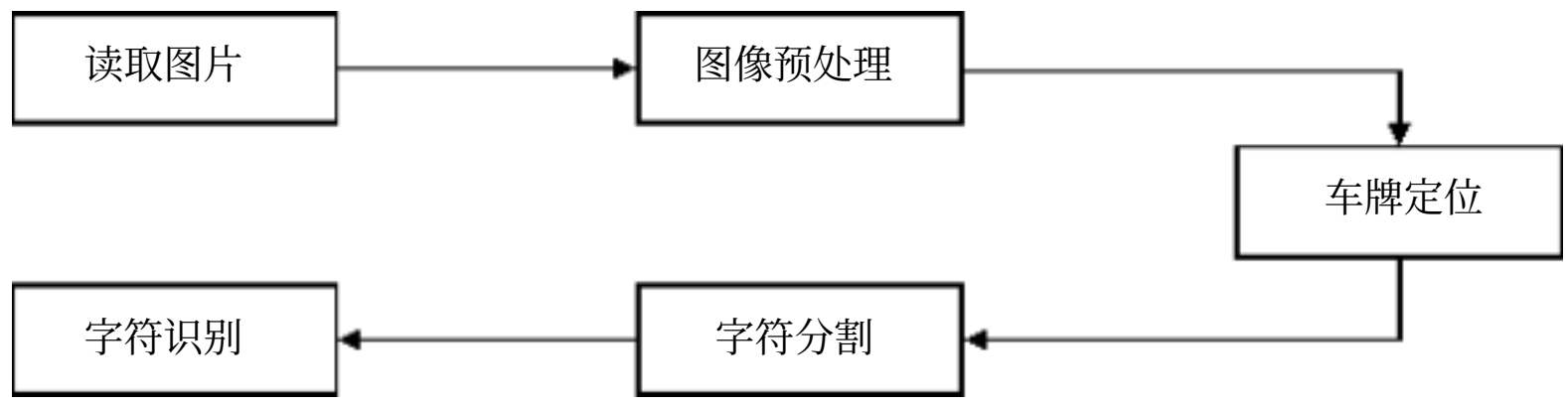


图 2.1　设计流程框图

2.1.1　系统功能模块分析

MATLAB 车牌识别系统主要包括图像采集、图像预处理、车牌定位、字符分割、字符识别五个关键环节，其基本工作模块如下：

（1）图像采集：使用采样精度高的照相机拍摄来获取图像。

（2）图像预处理：先将图像转换成便于计算机处理的二值化图像，通过灰度化、图像增强、边缘提取、二值化操作来完成。

（3）车牌定位：利用图像中车牌的边缘、形状等特征，再结合 Roberts 算子边缘检测、数字图像、形态学技术对车牌区域进行定位。

（4）字符分割：在二值化后的车牌图像中寻找连续有文字的块，切割出长度大于设定的阈值区域。

（5）字符识别：运用模板匹配算法，将分割后的得到的二值化字符尺寸大小缩小为模板库中字符的大小，然后与所有的模板进行匹配，得到相减数值最小的则为识别结果输出，并存储数据。

2.1.2　本系统拟采用方案

本设计的研究对象的是数码相机拍摄的蓝底白字车牌图像，图像预处理是将图像经过灰度化、图像增强、边缘提取二值化等操作转化成便于定位的二值化图像。图像定位是采用基于边缘检测、形态学处理的算法，定位精度高。字符分割采用的方法是在二值化后的车牌部分寻找连续有文字的块，大于设定阈值的长度部分则切割。由于基于颜色特征的车牌号码定位易受外界光照和环境色彩影响以及车本身颜色为蓝色的限制所以本系统定位采用纹理特征。在字符识别中，由于汉字相对于数字的复杂性，所以本设计模板库存放了 5 个汉字，26 个大写英文字母，10 个阿拉伯数字。

3　系统设计

通过以上对于车牌识别系统的详细分析，　本文设计的车牌识别系统是由五个模块来实现的，下面我将分别对组成系统的五个模块设计和系统整体设计进行详细介绍。

3.1　图像采集

车牌号码识别系统图片采集主要是用照相机拍摄所得。　确保拍摄图片清晰，　车牌完整，字符清晰，拍摄的角度越小车牌在平面图像中的变形越小，　识别的效果也越好。

目前我国车牌主要有黄底黑字，　蓝底白字，白底黑字和黑底白字。　为确保设计结果的准确性，　本设计主要以蓝底白字的车牌为研究对象。　本设计所采用的图片统一采用*.JPG 的格式。图　3.1 为所采集的图像。

图 3.1　所采集图像

3.2　图像预处理

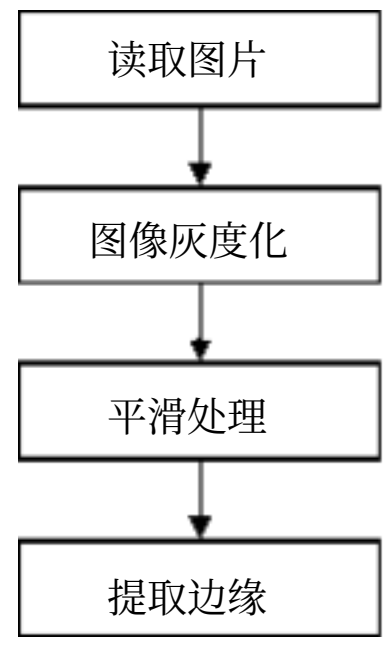


图 3.2　图像预处理基本步骤流程图

所拍摄图像由于比较大， 计算机处理起来比较慢， 为了能更快更准的进行汽车车牌的定位和分割。 为了能够准确的进行汽车车牌的定位和分割， 车牌定位之前必须经过图像预处理，预处理可以提高车牌图片的适用性 [9]。原始图像因为包括环境还有汽车本身及其他背景， 在处理过程中极易影响识别结果， 因此必须将干扰降至最小， 才有可能正确提取出汽车牌照所在区域。图像预处理的基本步骤如图 3.2 所示。

3.2.1 图像灰度化

原始图片包含的大量的颜色信息， 不仅占用了大量的存世内存， 而且会降低系统运行过程中的处理速度不但占用了很大存储内存， 而且在运行过程中也会降低系统的速度，图像灰度化 [10] 可以将彩色图片转换为灰度图片， 经转换后的图片只包含亮度信息，像素的动态范围增加，对比度增强，图像变得更细腻，清晰，容易识别。

彩色图像又称为 RGB 图像 [11]，它利用 R、G、B 三个分量表示一个像素的颜色，分别代表红、绿、蓝等三种颜色，灰度化过程就是使 R、G、B 三个分量相等的过程。因此对于一个尺寸为 M N 大小的彩色图像， 存储该图像就是存储一个 M N 3 的三位数组。在 RGB 模型中， R=G=B 的颜色表示一种灰度颜色。 R=G=B 的值就叫做灰度值， 图像灰度化就是将彩色图片转换为灰度图片。 灰度图像只包含亮度信息， 而没有颜色信息的。 存储灰度图像只需要存储一个二维的数组， 数组的每个元素表示对应像素点的灰度值。彩色图的像素色为 RGB（R， G， B），灰度图的像素色为 RGB（r，r，r），其中 R、G、B 可由彩色图中颜色分解得。 R、G、B 三个分量的取值范围是 0-255，因此灰度的级别只有 256 级。灰度化主要是利用加权平均值法：根据指标重要性的不同赋予 R、G、B 不同的权值，使得 R、G、B 等于它们的值的加权平均值。即：

$$R \quad G \quad B \quad (R \quad W_R \quad G \quad W_G \quad B \quad W_B)/3 \qquad \text{式（3.1）}$$

其中 W_R， W_G， W_B 分别对应 R、G、B 的权值，研究表明，由于人眼对绿色红色蓝色的敏感度由高到低，当 W_R 0.30， W_G 0.59， W_B 0.11 时，可得到最合理的灰度图像。 灰度其实就是亮度（ luminance）的量化值，而 RGB 的定义是客观的三个波长值，考虑到人眼对不同波长的灵敏度曲线，所以转化时系数不相等。

本设计的图片灰度变换调用的是 rgb2gray 函数， Gray 图中每个象素点的颜色都可以用直线 R=G=B 上的一个点来表示。 于是 rgb 图转 gray 图的原理就是确定一个由

三维空间到一维空间的映射，最常见的就是通过射影（即过 rgb 空间中的一个点向直线 R=G=B 做垂线），rgb2gray 正是通过这种方法。即：

$$Gray = 0.299R + 0.587G + 0.114B \quad \text{式（3.2）}$$

原始图像、灰度图和灰度直方图如图 3.3、图 3.4 所示。

图 3.3 灰度图

图 3.4 灰度直方图

3.2.2 图像增强

图像增强目的是对图像进行加工，从而更能准确进行车牌边缘检测。图像经过灰度化处理后 [12]，车牌部分和非车牌部分对比度并不是很高，车牌界限也较为模糊，不利于车牌边缘提取。图像增强便可加强图像中车牌和非车牌区域对比度，使其明暗鲜明，利于边缘检测。

图像增强的方法有许多，如灰度变换、线性滤波和图像平滑处理等，根据所处理图像域又可分为频域增强与空间增强。目前用于车牌号码识别的图像增强方法有：

（1）灰度拉伸：是类似于灰度的线性变换，虽然都用到灰度的线性变换，但灰度拉伸是分段进行线性变换。如图 3.5 所示。

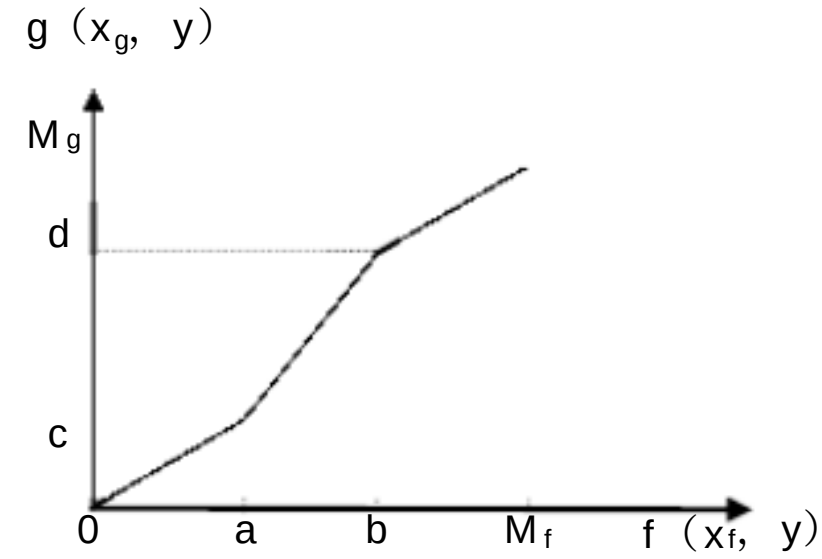


图 3.5　灰度拉伸示意图

灰度拉伸能更加灵活的控制所输出灰度直方图的分布，　它可以自主选择某段灰度区间并拉伸来改善输出图像。如图　3.5 所示，通过函数的运算，将原图中在　a 至 b 区间的灰度拉伸到　c 至 d 区间。若一幅图像的灰度集中在较暗的区域使得图像偏暗，　便可以采用灰度拉伸，通过拉伸斜率　>1 的物体灰度区间来改善图像；同理若图像灰度集中在较亮区域使得图像偏亮，也可以采用灰度拉伸，通过压缩斜率　<1 的物体灰度区间来改善图像质量。

灰度拉伸的原理与算法：

当 x<x1：　　f(x) = y1/x1*x;

当 x1<=x<=x2 ：　f(x) = (y2-y1)/(x2-x1)*(x-x1)+y1;

当 x>x2：　　f(x) = (255-y2)/(255-x2)*(x-x2)+y2;

//其中 x1,y1,x2,y2是图中 ac,bd 两个转折点的坐标。

```
void GrayStretch(BYTE X1, BYTE Y1, BYTE X2, BYTE Y2,BYTE *
grayMap)< xmlnamespace prefix ="o" ns ="urn:schemas-microsoft-com:office:office"
/>
{
    int x;
    grayMap[0]=0;
    for(x=1;x<=X1;x++)
    {
```

```
        grayMap[x]=(BYTE)x*Y1/X1;
    }
    for(;x<X2;x++)
    {
        grayMap[x]=(x-X1)*(Y2-Y1)/(X2-X1)+Y1;
    }
        grayMap[X2]=Y2;
        for(x++;x<255;x++)
        {
            grayMap[x]=(x-X2)*(255-Y2)/(255-X2)+Y2;
        }
        grayMap[255]=255;
    }
```

（2）直方图均衡化：直方图均衡化算法可分为三个步骤。 a.统计直方图中每个灰度级的出现次数， b.累计归一化的直方图， c.计算新的像素值。

```
第一步：
for(i=0;i<height;i++)
{
    for(j=0;j<width;j++)
    {
        n[s[i][j]]++;
    }
}
for(i=0;i<L;i++)
    {
        p[i]=n[i]/(width*height);
    }
```

其中， n[i] 表示灰度级为 i 的像素的个数， L 表示最大灰度级， width 表示原始图像的宽度， height 表示原始图像的高度，因此 p[i] 就表示灰度级为 i 的像素在整个图像中出现的概率 (p[] 这个数组存储的就是整幅图像归一化之后得到的直方图)。

第二步：

```
for(i=0;i<=L;i++)
{
    for(j=0;j<=i;j++)
    {
        c[i]+=p[j];
    }
}
```

c[]这个数组存储的就是累计的归一化直方图。

第三步：

```
max=min=s[0][0];
for(i=0;i<height;i++)
{
    for(j=0;j<width;j++)
    {
        if(max<s[i][j])
        {
            max=s[i][j];
        }
        else if(min>s[i][j])
        {
            min=s[i][j];
        }
}
}
```

找出像素的最大和最小值。

```
for(i=0;i<height;i++)
{
    for(j=0;j<width;j++)
    {
```

```
            t[i][j]=c[s[i][j]]*(max-min)+min;
        }
    }
```

最终直方图均衡化之后的结果就是 t[][] 。

（3）图像腐蚀 :采用 3 3 的结构元素，扫描图像中的每一个像素，用结构元素与扫描所覆盖的二值图像做 “ 与 ” 运算，如果都为 1，则像素为 1。否则置 0。处理后二值图像缩小一半。

（4）图像膨胀：与图像腐蚀相反，采用 3 3的结构元素，扫描图像中的每一个像素，用结构元素与扫描所覆盖的二值图像做 “ 与 ” 运算，如果都为 0，则像素为 0。否则置 1。处理后使二值图像扩大一圈。

考虑到车牌样本的具体情况， 本系统图像增强时先腐蚀后膨胀， 用来消除小的干扰域、在纤细点处分离物体与平滑较大物体的边缘同时并不改变其面积。 腐蚀后的图像如图 3.6 所示。

图 3.6 腐蚀后图像

在图像增强步骤中对于一些车牌字符明暗对比不明显的图像我们也可以采用以下方法进行图像的亮度调整：

（1）首先统计灰度图像的像素点，确定 V_1 和 V_2 两个值，使得亮度值 $x \in [min, V_1]$ 的像素数量占整个图像像素数量的 5%；$x \in [V_2, max]$ 的像素数量占整个图像像素数量的 5%。其中 min 表示图像中像素亮度的最小值， max 表示图像中像素亮度的最大值。

（2）然后我们将亮度值低于 V_1 的像素值置为 V_1，同样的，把亮度值高于 V_2 的像素值置为 V_2。这样我们就去除了个别特别暗或者特别亮的点。

（3）最后按照对应的比例将图像亮度从 [V 1，V 2]拉伸到 [0，255]，这样就完成了图像的亮度调整。完成图像亮度矫正之后， 可以通过图像的滤波来去除图像中的噪声。如图 3.7、图 3.8 所示。

图 3.7 1.7 倍亮度图像图

图 3.8 0.7 倍亮度图像

3.2.3 Roberts 边缘检测

车牌识别系统中的边缘是指图像中梯度方向上发生突变或者灰度值发生空间突变的像素的集合。 对图像进行边缘检测处理， 可以缩小检测范围， 便于后续的车牌区域和字符区域判定。 良好的边缘检测可以大大的降低图像中的噪声、 分离出复杂环境中所拍摄出的车牌图像、保留完好的车牌字符信息，为识别的准确率提供保障。

常用的图像边缘检测的算法有 Roberts 算子、 Sobel 算子和拉普拉斯高斯算子。以下是三种边缘检测算子的特点：

（1）Sobel 算子：它是方向性的，在水平和垂直方向上形成了最明显的边缘。 Sobel 算子不仅能检测出边缘点，而且能抵制噪声影响， Sobel 算子已在数学上证明当像素点分布满足正态分布时，边缘检测是最优的。

该算子包含两组横向和纵向的 3×3 矩阵，将矩阵与图像作平面卷积， 便可以，分别得出横向的亮度差分近似值和纵向的差分近似值。若以 A 代表原始图像， Gx 代表经横向， Gy 代表纵向边缘检测的图像，公式如下：

$$G_x = \begin{bmatrix} 1 & 0 & 1 \\ 2 & 0 & 2 \\ 1 & 0 & 1 \end{bmatrix} A \quad \text{and} \quad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} A \qquad \text{式（3.3）}$$

图像中每一个像素的横向和纵向梯度近似值可通过以下公式来计算：

$$G \quad \sqrt{G_x^{\,2} \quad G_y^{\,2}} \qquad \text{式（3.4）}$$

然后通过以下公式计算梯度方向：

$$\theta \quad \arctan \frac{G_y}{G_x} \qquad \text{式（3.5）}$$

以上所示，若角度 θ 0，即表示图像在该处有纵向的边缘，左方比右方暗。

（2）Roberts算子：边缘检测比较准确，但极易受图像中噪声的干扰，去噪声能力差，适用于边缘明显且噪声较小的图像。 Roberts算子的原理公式如下：

$$g\ x,y \quad \sqrt{f\ x,y} \quad \sqrt{f\ x \quad 1,y \quad 1}^{\,2} \quad \sqrt{f(x \quad 1,y)} \quad \sqrt{f\ x,y \quad 1}^{\,2} \qquad \text{式（3.6）}$$

其中 f（x，y）、f（x+1，y）、f（x，y+1）和 f（x+1，y+1）分别为 4 个邻域的坐标，且是具有整数像素坐标的图像。 平方根运算处理类似于人类视觉系统中发生的过程。

Roberts 算子采用的是 2 2 的算子模板，包含两个卷积核，图像中每个像素点都用这 2 个核做卷积。如图 3.9 所示。

1	0
0	-1

0	1
-1	0

图 3.9 Roberts 算子模板

（3）Prewitt 算子：是一种基于一阶微分算子的边缘检测，利用某个点像素点四周相邻点的灰度差，在边缘处达到极值，删除一些伪边缘，可使噪声变得平滑。它的原理是在图像空间中采用不同方向的两个模板与待处理图像进行邻域卷积， 这两个模板一个检测水平边缘，另一个检测垂直边缘。如图 3.10 所示。

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

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

图 3.10 两个方向模板

对于数字图像 f(x， y)， Prewitt 算子的定义如下：

$$G\ i \quad \left| f\ i-1,j-1 \quad f\ i-1,j \quad f\ i-1,j \quad 1 \ - f\ i \quad 1,j-1 \quad f\ i \quad 1,j \quad f\ i \quad 1,j \quad 1 \right| \qquad \text{式（3.7）}$$

$$G_i = |f_{i-1,j+1} + f_{i,j+1} + f_{i+1,j+1} - f_{i-1,j-1} + f_{i,j-1} + f_{i+1,j-1}| \quad \text{式（3.8）}$$

得 $P_{i,j} = \max(G_i, G_j)$ 或 $P_{i,j} = G_i + G_j$

经典 Prewitt 算子：凡是灰度新值 阈值的像素点均为边缘点。即确定合适的阈值 T，若 P(i,j) T≥ 则点 (i,j) 为边缘点， P(i,j) 为边缘图像。事实证明这种判定是不合理的，由于许多噪声点的灰度值很大，对于幅度较小的边缘点，会丢失其边缘，从而造成边缘点的误判。

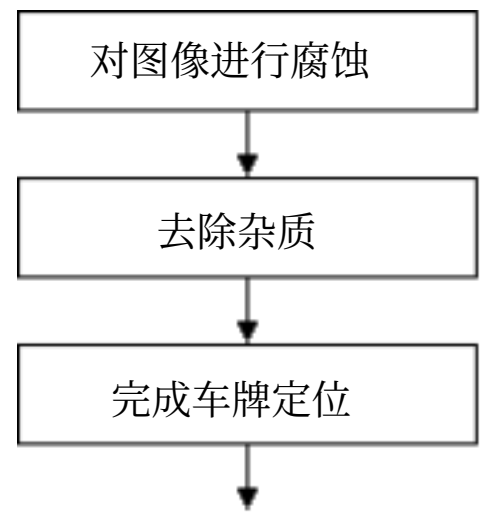
三种算子边缘检测的对比如图 3.11 所示。

图 3.11 从左往右依次为 Sobel、 Roberts、 Prewitt 算子边缘检测

本设计是用 Roberts 算子实现边缘检测的，实际处理过程中边缘检测结果如图 3.12 所示。

3.12 Roberts 边缘检测图像

3.3 定位和分割



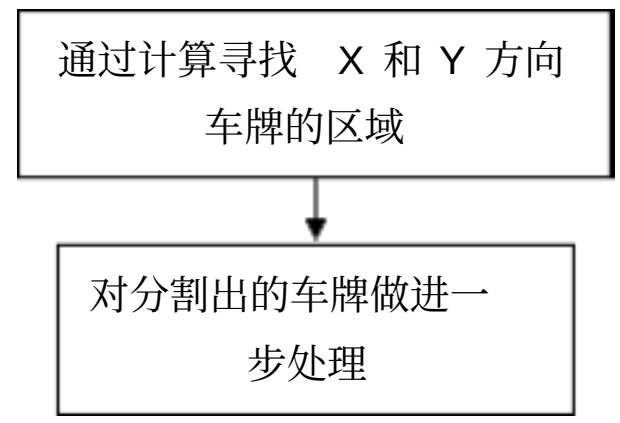


图 3.13 车牌定位流程图

车牌号码的定位和分割是汽车号牌识别系统的核心技术之一， 车牌号码定位和分割的准确与否直接关系到最终识别的结果 [13]。汽车车牌是一个很有特征的区域， 近似一个长方形， 其大块的蓝色区域灰度值与周围区域有着明显的不同， 所以在车牌边缘形成了灰度突变的边界， 这样就便于通过边缘检测来对图像进行分割。 定位的流程图 3.13 所示。

3.3.1 车牌定位

如何在复杂背景中准确、 快速找出车牌的位置。 车牌定位技术需要克服图像环境复杂多变，车牌自身磨损，沾上污渍等不利因素。另外，由于拍摄角度的不同，车牌在图像中造成的形变情况也不同 [14]。

目前常用的定位主要有如下几类方法：

（1）基于水平灰度变化特征的方法，该方法在车牌定位前，需要将彩色图像转换 为灰度图像的预处理步骤，然后利用车牌区域在水平方向的纹理特征进行车牌定位；

（2）基于边缘检测的定位方法，该方法是利用车牌区域丰富的边缘特征进行车牌定位，进行边缘检测的算法有： Roberts 边缘算子、 Prewitt 算子、 Sobel 算子以及拉普拉斯边缘检测；

（3）基于颜色特征的定位方法，该方法主要是利用车牌字符和车牌底色具有明显的反差特征来排除干扰区域进行车牌的定位；

（4）基于 Hough 变换的车牌定位方法，该方法是利用车牌边框的几何特征，采取寻找车牌长方形边框直线的方法进行车牌定位；

（5）基于变换域的车牌定位方法，该方法通过将图像从空域变换到频域进行分析，如采用小波变换等；

（6）基于数学形态学的车牌定位方法，顾名思义该方法是基于数学形态学图像处理的基本思想。 利用一个结构元素来探测图像， 看是否能将这个结构元素很好的填

放入图像内部，并验证填放元素的方法是否有效。数学形态学的基本运算有：腐蚀、膨胀、开启和关闭。

以上介绍的方法各有优缺点，　要快速、准确地定位车牌，　应该综合考虑车牌图像的多种特征。本系统设计从实际出发结合车牌的纹理、颜色、水平灰度变化特征、采用Roberts 算子边缘检测和数学形态学的腐蚀、膨胀等特征对车牌进行了定位，对于提高车牌定位准确率提供更有利的保障。下面是车牌定位的相关图片及代码：

```
se=[1;1;1];
I3=imerode(I2,se);
guidata(hObject, handles);
se=strel('rectangle',[10,25]);
I4=imclose(I3,se);
guidata(hObject, handles);
I5=bwareaopen(I4,2000);
guidata(hObject, handles);
function [PY2,PY1,PX2,PX1]=chepai_fenge(I5)
[y,x,z]=size(I5);
myI=double(I5);
tic
Y_threshlow=5;
X_firrectify=5;
Blue_y=zeros(y,1);
for i=1:y
    for j=1:x
        if(myI(i,j,1)==1)
            Blue_y(i,1)= Blue_y(i,1)+1;
        end
    end
end
[temp MaxY]=max(Blue_y);
PY1=MaxY;
```

```
while ((Blue_y(PY1,1)>=Y_threshlow)&&(PY1>1))
        PY1=PY1-1;
end
        PY2=MaxY;
while ((Blue_y(PY2,1)>=Y_threshlow)&&(PY2<y))
        PY2=PY2+1;
end
        PY1, PY2
figure(1),imshow(Blue_y),title('y 方向确定 ');
```

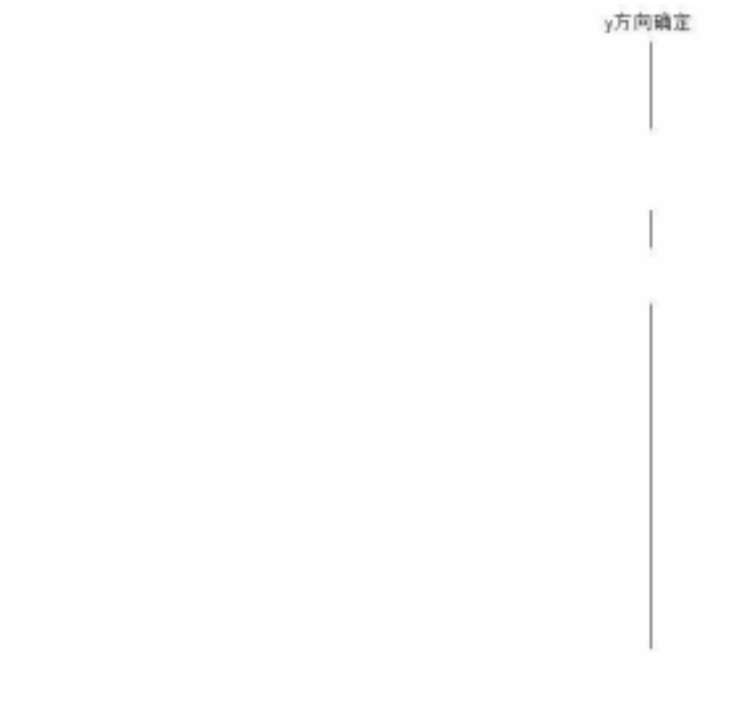


图 3.14　y方向区域确定

```
PX1=PX1-1; PX2=PX2+1;
dw=I(PY1:PY2-8,PX1:PX2);
imshow(dw),title( 水平方向合理区域 ’);
pause(2);
t=toc;
guidata(hObject, handles);
```

图3.15　水平方向定位区域

3.3.2　车牌分割

车牌分割的方法有多种多样，由于蓝色车牌的颜色信息与其他背景区别比较大，所以本系统采用彩色分割的方法。　本系统设计研究的对象车牌底色是蓝色。　统计蓝色像素点，分割出合理的车牌区域。明确车牌底色蓝色　　RGB 对应的各自灰度范围，然

后设定合理的阈值， 分别从行方向与列方向统计在此颜色范围内的像素点数量， 确定车牌在水平方向的合理区域， 最终确定出完整的车牌区域。 以下是车牌分割的关键程序说明以及效果图：

```
myI=double(I5);%begin横向扫描
tic
Y_threshlow=5; %这个数值很重要。决定了提取的彩图的质量
X_firrectify=5;%=== Y 方向===从左向右寻找第一个 1值像素大于 5的坐标为水平方向左侧分界线，从优向左寻找到第一个 1值像素量大于 5的为右侧分界线
Blue_y=zeros(y,1);
for i=1:y
    for j=1:x
        if(myI(i,j,1)==1)% 如果myI(i,j,1) 即myI图像中坐标为 (i,j) 的点为白色
                         %则Blue_y的相应行的元素 white_y(i,1) 值加 1
            Blue_y(i,1)= Blue_y(i,1)+1;          % 蓝色象素点统计
        end
    end
end
[temp MaxY]=max(Blue_y);% Y 方向车牌区域确定 temp(最多点数 ):所有行中， 最多的累积像素点 MaxY （最多点所在行） :该行中蓝点最多
PY1=MaxY;% 有最多蓝点的行付给 PY1
while ((Blue_y(PY1,1)>=Y_threshlow)&&(PY1>1))% 找到图片上边界
    PY1=PY1-1;
end
%PY1：存储车牌上边界值
PY2=MaxY;
while ((Blue_y(PY2,1)>=Y_threshlow)&&(PY2<y))% 阈值为 5
    PY2=PY2+1;
end
PY1, PY2 %原始图像 I中截取的纵坐标在 PY1：PY2之间的部分
figure(1),imshow(Blue_y),title('y 方向确定 ');
```

```
pause(2);
%==============X  方向================================
X_threshhigh=(PY2-PY1)/11;%这个数值很重要。决定了提取的彩图的质量    ,适当提高可抗干扰，但是小图会照成剪裁太多
Blue_x=zeros(1,x);% 进一步确定 X方向的车牌区域
for j=1:x
    for i=PY1:PY2
        if(myI(i,j,1)==1)
          Blue_x(1,j)= Blue_x(1,j)+1;
        end
    end
end
[temp MaxX]=max(Blue_x);
PX1=MaxX-6*(PY2-PY1);
if   PX1<=1
    PX1=1;
end
while ((Blue_x(1,PX1)<=X_threshhigh)&&(PX1<x))%  阈值
        PX1=PX1+1;
end%确定出X方向车牌起点
PX2=MaxX+6*(PY2-PY1);
if   PX2>=x
    PX2=x;
end
while ((Blue_x(1,PX2)<=X_threshhigh)&&(PX2>PX1))%  阈值
        PX2=PX2-1;
end%确定出X方向车牌终点
PX1 ,PX2
figure(2),imshow(Blue_x),title('X 方向确定 ');
```

图 3.16　分割后的车牌

3.3.3　车牌进一步处理

在图像处理过程中，二值化是至关重要的一步，图像的二值化能使使图像变得简单，后续所需处理的量减小，能使目标轮廓变得更加清晰。二值化图像中整个图像只有黑白两中颜色。二值化算法的原理比较简单，即确定一个阈值，将灰度图像中像素的亮度值小于该阈值时，便将该像素值置 0；而当灰度图像中某像素的亮度值大于等于该阈值时，就将该像素值设为 1，如此便完成了图像的二值化。关于阈值如何选取，有三种方法：全局阈值法、局部阈值法和动态阈值法。常见的算法有 Ostu 法和熵函数法[15]，但运算时间都比较长。因此，通常采用平均值法进行二值化[16]，设阈值为 TH 。如式（3.9）所示。

$$TH = \frac{1}{N}\sum_{i=0}^{X-1}\sum_{j=0}^{Y-1} G(i,j) \qquad \text{式（3.9）}$$

其中 G（i，j）为像素点的灰度值，X 为图像宽度，Y 为图像高度。系数 N 根据三种情况确定：

（1）整幅图像背景几近无干扰，均值很小；

（2）整幅图像背景有干扰，均值较大；

（3）整幅图像光照很暗，均值中等。

车牌分割出来后，其图像背景中仍存在噪声，想要直接提取出所需目标，最常用的方法就是对图像二值化。还有可采用均值滤波的方法，均值滤波是典型的线性滤波算法，它的原理是对目标像素给一个模板，该模板包含了周围相邻的像素。然后用模板中所有像素的均值替换原来的像素。车牌腐蚀原理同上文图像增强时的图像腐蚀原理。车牌图像处理过的图如图 3.17 所示。

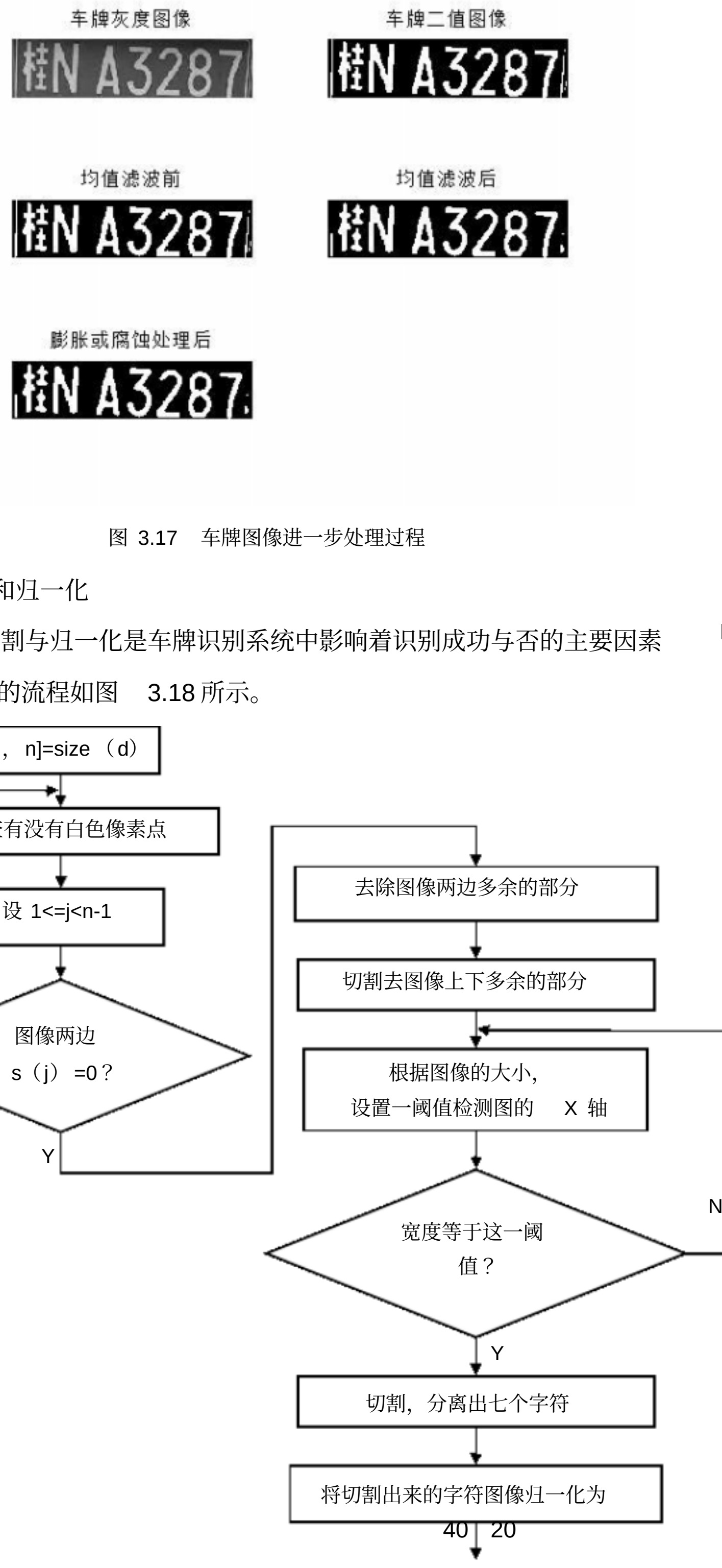


图 3.17　车牌图像进一步处理过程

3.3.4　字符分割和归一化

车牌字符的分割与归一化是车牌识别系统中影响着识别成功与否的主要因素[17]。字符分割与归一化的流程如图 3.18 所示。

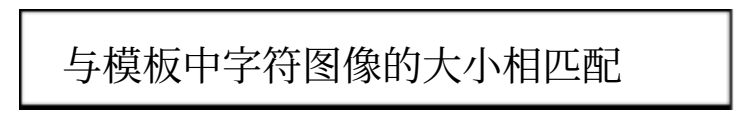


图 3.18　车牌图像处理过程

3.3.5　字符分割

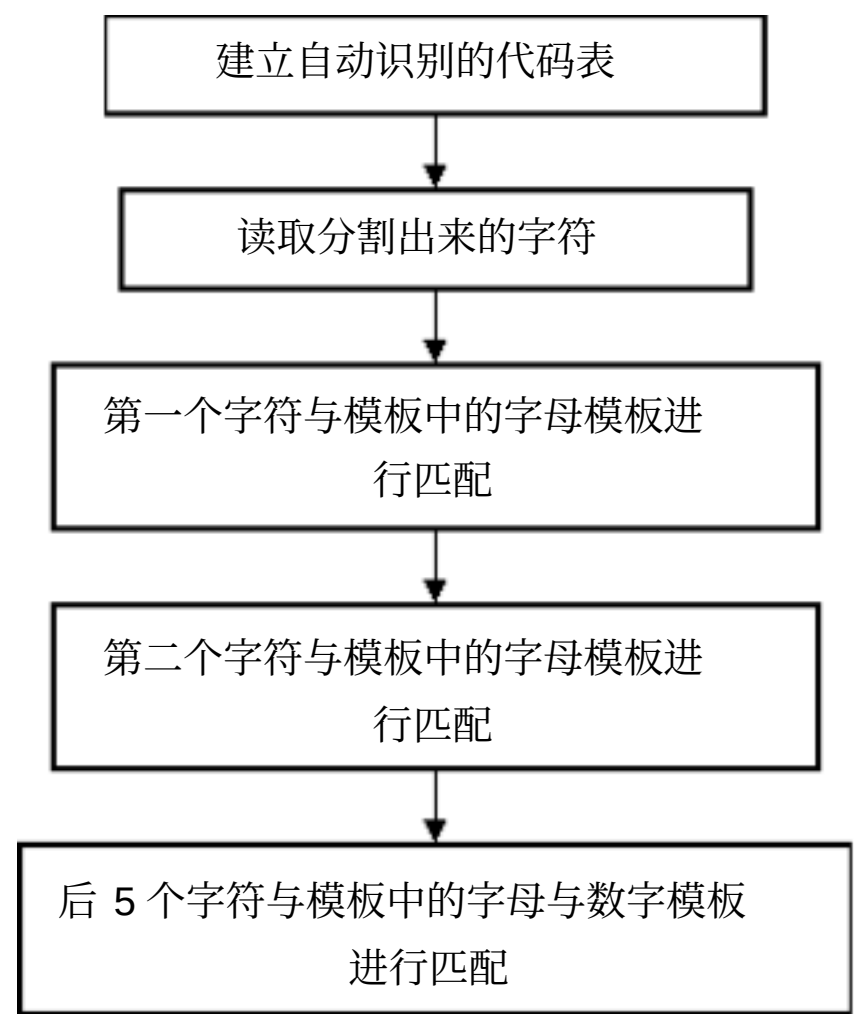
在车牌号码识别过程中，字符识别是前期车牌定位处理后，利用分割的结果，最终完成识别必不可少的一步。车牌字符之间空隙较大，易于分割提取，所以我采用的方法是设定一个阈值，然后寻找连续的像素部分，寻找到长度大于阈值的部分，则判定该区域含有两个字符，并进行分割。字符分割的结果如图 3.19 所示。

图 3.19　字符分割的结果

3.3.6　字符归一化

字符分割后，字符图像会存在一定的大小差距，不加以处理无法进行后续字符识别中的特征提取 [18]。因此我对分割出来后的字符进行了归一化处理，即将字符大小统一设置为 40×20。

3.4　字符识别



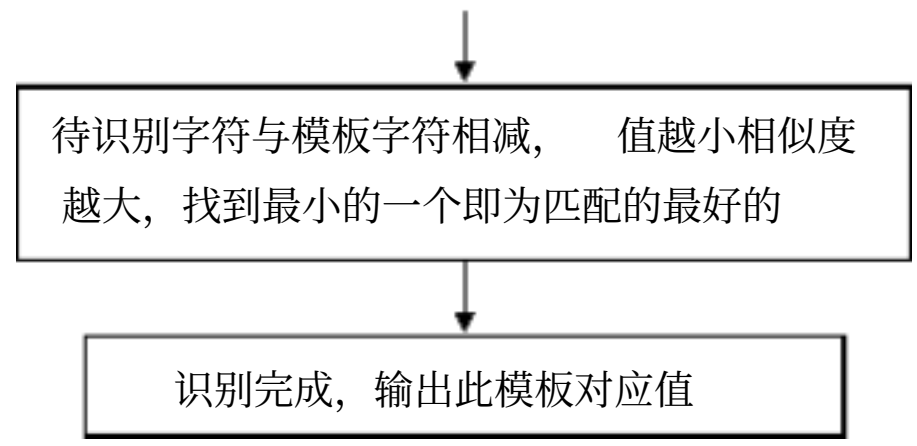


图 3.20　模式识别流程图

目前主要的字符识别方案有　4 种：模板匹配字符识别算法、　统计特征匹配法、　神经网络字符识别算法和支持向量机模式识别算法。　其中，模板匹配原理和步骤相对简单，神经网络字符识别算法是目前比较主流的算法，　以下我简要介绍一下本系统所研究的模版匹配算法和神经网络识别算法：

（1）模板匹配字符识别算法 [19]：模板匹配算法的原理是，通过计算比较输入模板与样本之间的相似程度，　最终取相似程度最大的为输入模板的类别。　这种方法处理速度快，但易受噪点的影响。在实际应用中往往需要大量的模板或多个模板进行比对，才能保证一定的准确率。

（2）神经网络字符识别算法主要分两种 [20]：一是对待侧字符样本进行特征的提取，接着通过所获得的特征来训练神经网络分类器。　这种算法的关键是字符特征的提取，训练的时间随着特征参数的多少来决定。　参数过少会不利于识别的准确率。　第二种方法是充分利用神经网络的特点，　直接输入待处理图像到网络，　由网络自动实现特征提取并完成字符识别。　这种算法的神经网络网络互连较多，　处理信息量大，　能很好的抗干扰，识别率也高。但是网络结构十分复杂，因为输入模式维数很多，造成网络规模巨大。当车牌字符完整，图像清晰，无干扰时识别率相当高。其流程图如图　3.20 所示。

在本设计模式字符识别方案中采用的是归一化后字符样本与所设模板相减的方法找出样本与模板库中相似程度最大的字符，　然后输出此模板的对应值。　汽车号牌的字符数是七个。通常第一位为汉字，接着是英文字符和数字。车牌字符汉字共约　50 个，大写英文字符　26 个，数字　10 个。为了减少不必要的工作量，本设计模板库只纳入了 5 个汉字 26 个大写英文字母与　10 个数字。字符识别图如图　3.21 所示。

图 3.21　车牌号码识别结果

3.5　模式识别系统结果与分析

3.5.1　结果

以下是我所统计的模式识别结果如表　3.1、3.2 所示：

表 3.1　汉字识别结果

字符	苏	京	鲁	豫	桂
苏			1		
京			1		
鲁			1		
豫			1		
桂			3		5

表中竖列表示车牌字符，横列表示识别结果。中间数字为识别出该结果的次数。

由此可见汉字的误码率为：　$\frac{1\ \ 1\ \ 1\ \ 3}{1\ \ 1\ \ 1\ \ 1\ \ 3\ \ 5}$　100%　50% 。

表 3.2　英文数字识别结果

字符	A	B	C	H	J	L	M	N	P	Q	0	1	2	3	5	6	7	8	9
A	7																		
B							1												
C											1								
H				1															
J					1														
L			2																
M							1												

N				1			6												
P									2										
Q							2												
0		3		4			4									1			
1							1												
2							2												
3		4					2												
5		1					1												
6		1														5			
7																	6		
8		2					4												
9							1												3

分析数据可发现英文和数字的误码率为： $\frac{46}{46 \quad 26}$ 100% 63.9% 。

3.5.2 分析

经以上实验结果可发现识别错误率大致集中在外形相似的字符，如 N 和 M，8 和 B， C 和 0 等，主要原因是单纯的模板相减，在字符与模板相似程度大的情况下，差值便不明显造成了误码。 另外模板的不清楚也极易造成误码， 因此要想进一步提高识别率，我总结了以下方面还需进一步的研究改进：

（1）车牌定位虽然采用了边缘检测和颜色特征提取，但抗干扰能力仍有进一步改

进的空间。在图像不够清晰或外界光照影响下， 定位不准的情况虽然不多但仍有发生。

（2）模板匹配字符识别算法法，方法简洁但识别率较低。模板库的字符制作很重

要，应尽量采用清晰准确的模板，否则易造成大量误码。

（3）单纯的一个字符只采用一个模板进行比对极容易受到其它相似模板的干扰，

要想避免这一点，采用大量的模板进行多次比对最终得出的值更容易接近正确值。

3.6 小结

以上我们讲述了模式识别从图像提取到最终完成字符识别的详细过程， 分析了处

理过程中各算子的优点与不足，　并探讨如何改进。 虽然前期处理直至字符分割的效果都还不错，但模式识别的结果并不理想。　模式识别的改进方向是对每个字符采用多个该字符模板进行比对，　从而降低其他字符模板的干扰，　这种方法与神经网络字符识别方法类似。接下来本设计将介绍所采用的神经网络字符识别算法。

4　人工神经网络字符识别

神经网络识别技术是目前主流的识别技术，　它较好的解决了车牌识别中因字符残缺导致无法识别的问题。 也可避免繁重的数学建模和数据分析工作，　并可将信息存储与处理并行， 大幅提高运行速度 [21]。目前研究使用最多的为前向反馈神经网络　（简称 BP 网络）。

4.1 BP　网络模型结构

BP 神经网络是一个单向传播的多层前向神经网络，它包含输入层，隐含层和输出层，层与层之间采用的是全连接方式，同一层之间不存在链接。　BP 神经网络的学习过程分为正向传播和方向传播两种交替过程。在正向传播中，输入信息从输入层，隐含层，经过逐层计算传向输出层，　单层神经元的状态只会影响到相连的下一个神经元状态。若在输出层没有得到希望的输出，　则通过计算输出层输出的误差变化值，　然后转向方向传播，通过网络将误差信号沿原来的通路回馈回来，　修改各层神经元权值，直至达到期望值。

下图为仅含一个隐含层的　BP 网络，输入层包含　r 个节点 x1、x2、？xr，隐含层 s1 个节点： y1、y2？ys1，隐含层对应的激活函数为　f1，输出层有 s2 个节点： z1、z2、？zs2，对应的激活函数为　f2，输出为 A，目标矢量为 T。BP 神经网络的结构如图 4.1 所示。

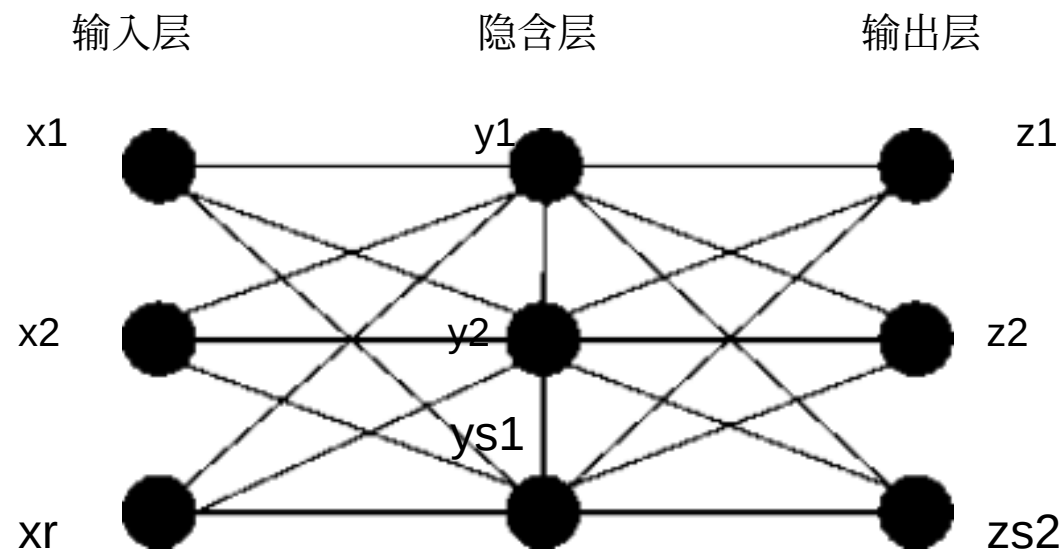


图 4.1　BP 神经网络的结构图

4.1.1　输入信息正向传递

隐含层中第　i 个神经元的输出为：

$$a1_i \quad f1 \quad \sum_{j\ 1}^{r} w1_{ij}\, p_j \quad b1_i \quad ,i \quad 1,2, \quad ,s1 \qquad \text{式（4.1）}$$

输出层第 k 个神经元的输出为：

$$a2_k = f2\left(\sum_{i=1}^{s1} w2_{ki}\, a1_i + b2_k\right), k = 1,2,\ldots,s2 \qquad \text{式（4.2）}$$

定义误差函数为：

$$E(W,B) = \frac{1}{2}\sum_{k=1}^{s2}(t_k - a2_k)^2 \qquad \text{式（4.3）}$$

4.1.2 利用梯度下降法求权值变化及误差的反向传播

（1）输出层的权值变化对从第 i 个输入到第 k 个输出的权值有：

$$\Delta w2_{ki} = -\eta\frac{\partial E}{\partial w2_{ki}} = -\eta\frac{\partial E}{\partial a2_k}\frac{\partial a2_k}{\partial w2_{ki}} = \eta(t_k - a2_k)f2'(a1_i) = \eta\,\delta_{ki}\,a1_i \qquad \text{式（4.4）}$$

其中：$\delta_{ki} = (t_k - a2_k)f2'(e_k) = f2'(e_k)(t_k - a2_k)$

同理可得从第 i 个输入到第 k 个输出的阈值：

$$\Delta b2_{ki} = -\eta\frac{\partial E}{\partial b2_{ki}} = -\eta\frac{\partial E}{\partial a2_k}\frac{\partial a2_k}{\partial b2_{ki}} = \eta(t_k - a2_k)f2_i'(\cdot) = \eta\,\delta_{ki} \qquad \text{式（4.5）}$$

（2）隐含层的权值变化

对从第 j 个输入到第 i 个输出的权值有：

$$\Delta w1_{ij} = -\eta\frac{\partial E}{\partial w1_{ij}} = -\eta\frac{\partial E}{\partial a2_k}\frac{\partial a2_k}{\partial a1_i}\frac{\partial a1_i}{\partial w1_{ij}} = \eta\sum_{k=1}^{s2}(t_k - a2_k)f2'(w2_{ki})f1'(p_j) = \eta\,\delta_{ij}\,p_j \qquad \text{式（4.6）}$$

其中：$\delta_{ij} = f1'(e_i)\sum_{k=1}^{s2}\delta_{ki}w2_{ki} = \delta_{ki}f2'(e_k) = f2'(e_k)(t_k - a2_k)$ 式（4.7）

同理可得从第 j 个输入到第 i 个输出的阈值：

$$\Delta b1_{ij} = \eta\,\delta_{ij} \qquad \text{式（4.8）}$$

以上式子中 η是步长调整的因子，即修正权值的学习速率： $0 < \eta < 1$。

4.2 BP 神经网络的训练

按照上述的权值修正公式，来进行神经网络的训练，当系统误差达到最小值时，神经网络趋于稳定，即学习结束。

BP 神经网络的训练过程为：

（1）将权值初始化为 0~1 之间的任意值；

（2）从样本中取出 x_0，x_1~x_{N-1}，输入网络，指定期望输出值 d_0，d_1~d_{M-1}；

（3）计算隐含层输出值 h_0，h_1~h_{L-1} 和网络实际输出 y_0，y_1~y_{M-1}；

（4）计算实际输出和期望输出的误差：

$$\delta_k \quad d_k \quad y_k \; y_k 1 \quad y_k \quad k \quad 1,2,M \quad 1 \qquad \text{式（4.9）}$$

计算隐含层误差：

$$\delta_j \quad h_j 1 \quad h_j \sum_{k\ 0}^{M\ 1} \delta_k w_{jk} \qquad \text{式（4.10）}$$

（5）调整权值：

$$w_{jk}^{\ n\ 1} \quad w_{jk}^{\ n} \quad \eta_k \delta h_j \qquad \text{式（4.11）}$$

$$v_{ji}^{\ n\ 1} \quad v_{ji}^{\ n} \quad \eta_j \delta x_i \qquad \text{式（4.12）}$$

式中 η 为学习因子。

（6）返回第（2）步，使用全部样本反复训练网络，通过多次迭代，直至达到稳定的权值。并且在实际训练中，定义出反映实际输出和期望误差平方和的价值函数：

$$E_p \quad \frac{1}{2} \sum_{K\ 0}^{M\ 1} d_k \quad y_k \ ^2 \qquad \text{式（4.13）}$$

定义收敛规则：

$$E \quad \frac{1}{p} \sum_{i\ 0}^{p\ 1} E_i \quad \varepsilon \qquad \text{式（4.14）}$$

式中 p 为训练的样本数，ε 为给定的误差范围，当满足 E<ε 时，则训练结束。经过训练后的网络就可以工作了，对某一输入使它认知，并进行识别。

4.3 BP 神经网络结构和设计

进行神经网络识别之前，首先要从为训练准备的识别字符库的样本中，提取能代表该字符的特征向量。再将所提取的特征向量输入 BP 网络之中就可以对网络进行训练，然后从待识别的字符样本中提取特征向量带入到训练好的 BP 网络中，便可以进行字符识别。

4.3.1 特征向量提取

本设计选取第 20 特征法，首先将字符平均分为 16 等份，并统计 16 份中白色像素点的比例，作为 16 个特征向量。如图 4.2 所示。

图 4.2　特征提取法

接着统计统计水平方向中间两块区域和竖直方向中间两块区域中的白色像素点比例作为后 4 个特征向量，最后统计所得的 20 个白色像素点比例为 20 个特征。如图 4.3 所示。

图 4.3　4 个特征示意图

4.3.2 BP 神经网络的结构与设计

根据车牌字符格式的特点：第一个字符为汉字，第二个字符为英文字母，第三、四个字符为英文字母或数字，第五、六、七字符均为数字。本设计构造了 4 个含一个隐含层的三层 BP 神经网络，分别用来识别汉字、英文字母、数字或英文字母、数字。识别汉字的 BP 神经网络，输入层有 20 个节点，输出层 2 个节点（“桂”、“苏”）。识别英文字母的 BP 神经网络，输入层有 20 个节点，输出层有 24 个节点（大写的英文 26 个字母 I 和 O 除外）。识别英文字母或数字的 BP 神经网络，输入层有 20 个节点，输出层有 34 个节点（大写英文字母 24 个，数字 10 个），识别数字的 BP 神经网络，输入层有 20 个节点，输出层 10 个节点（数字 10 个）。

4.3.3 初始权值、激活函数的选取以及各参数的设定

BP 神经网络中的权值和阈值为随机选取，输入层和隐含层的激活函数分别为 tansig 函数和 logsig 函数。各参数的设定为：误差目标 goal=0.0005；学习速度 lr=0.1；学习速率递增乘因子 lr_inc=1.15；学习速率递减乘因子 lr_dec=0.8；动量因子

net.trainParam.mc=0.9;

4.3.4 网络的训练

设计总共使用了 90 张汽车图片，其中选取 30 张图片，共 359 张车牌字符做为训练样本，部分样本如图 4.4 所示。

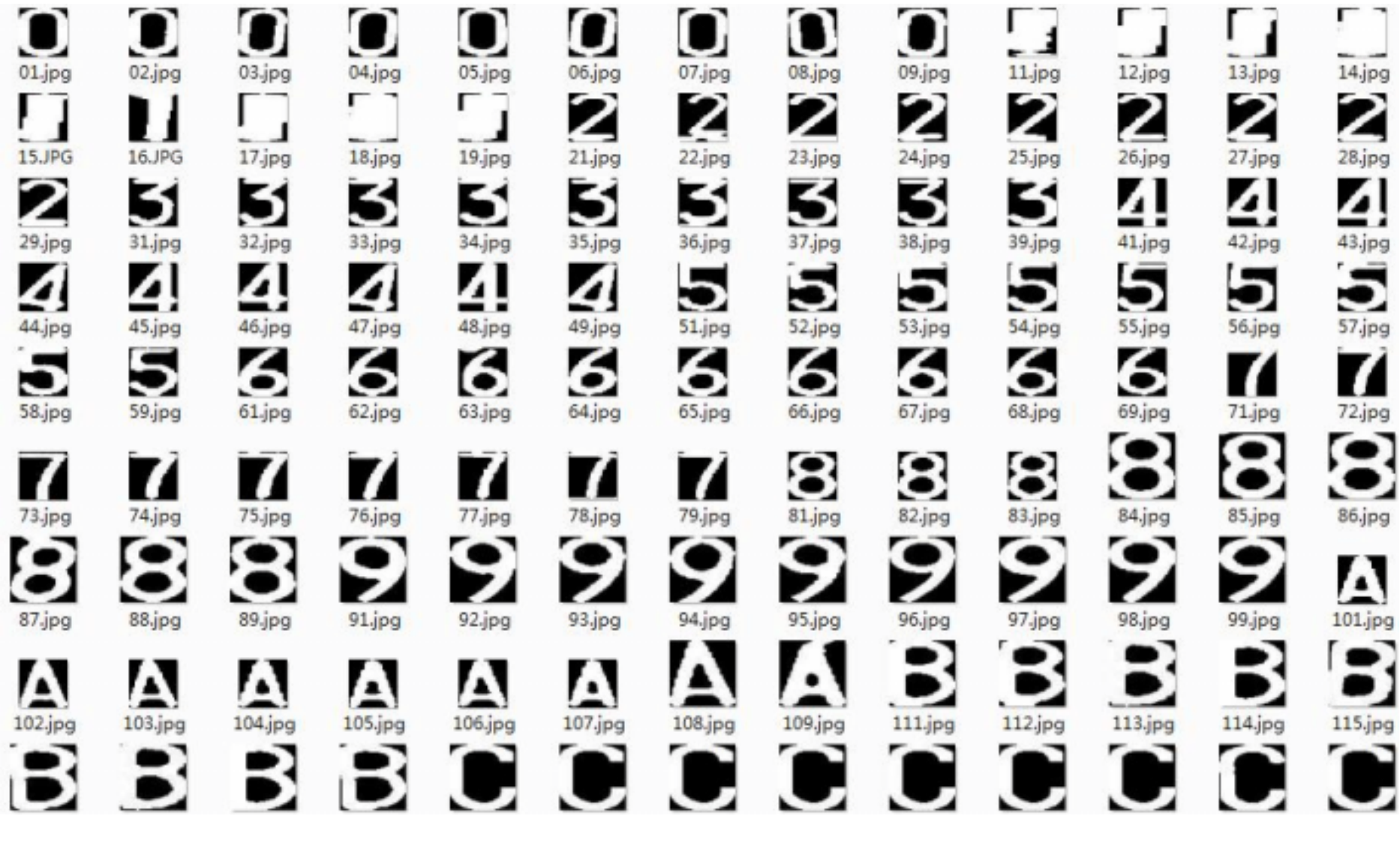


图 4.4 部分字符训练样本

其余的 60 张为测试样本。经过预处理，车牌定位，字符分割处理后，原图片变成大小为 20×40 的字符图像，如图 4.5 所示。接着采用 20 特征法进行特征提取，将所提取的特征作为网络的输入。 训练过程中每一个车牌字符， 选取 9 个字符样本进行训练。

图 4.5 分割后的车牌字符

训练参数的设定如下：

```
net=newff(minmax(P),[41,24],{'tansig','logsig'},'traingdx');%Kolmogorve 隐含层神经数为 2 a 1 a为输入个数 20
net.trainParam.show=50;          %显示间隔
net.trainParam.lr=0.05;          %学习速率
net.trainParam.lr_inc=1.15;      %学习速率递增乘因子
```

net.trainParam.lr_dec=0.8　　%学习速率递减乘因子

net.trainParam.mc=0.9　　%动量常数

net.trainParam.epochs=3000;　　%最多循环次数

net.trainParam.goal=0.005;

接下来就可以进行 BP网络的训练了：

[net,tr]=train(net,P,T);

以创建训练识别数字的 BP网络为例：

```
clear all
for y=1:9
  for x=0:35
    i0=imread(strcat('bp_train_images/',strcat(num2str(x),strcat(num2str(y),'.jpg'))));
    i0=i0>100;
    for i=1:16
      top=1+floor((i-1)/4)*16;
      buttom=top+15;
      left=1+rem(i-1,4)*16;
      right=left+15;
      i2=i0(top:buttom,left:right);
      s(i)=sum(sum(i2))/16/16;
    end
    s(17)=sum(s(5:8))/4;
    s(18)=sum(s(9:12))/4;
    s(19)=(s(2)+s(6)+s(10)+s(14))/4;
    s(20)=(s(3)+s(7)+s(11)+s(15))/4;
    if x<=9
        ar_num(:,x+1+10*(y-1))=s;
        end
    if x<=33
        ar_num_alp(:,x+1+34*(y-1))=s;
        end
```

```
if x>9&&x<34
    ar_alp(:,x-9+24*(y-1))=s;
    end
if x>33
    ar_char(:,x-33+2*(y-1))=s;
    end
  end
end
```

训练结果如图 4.6 所示。

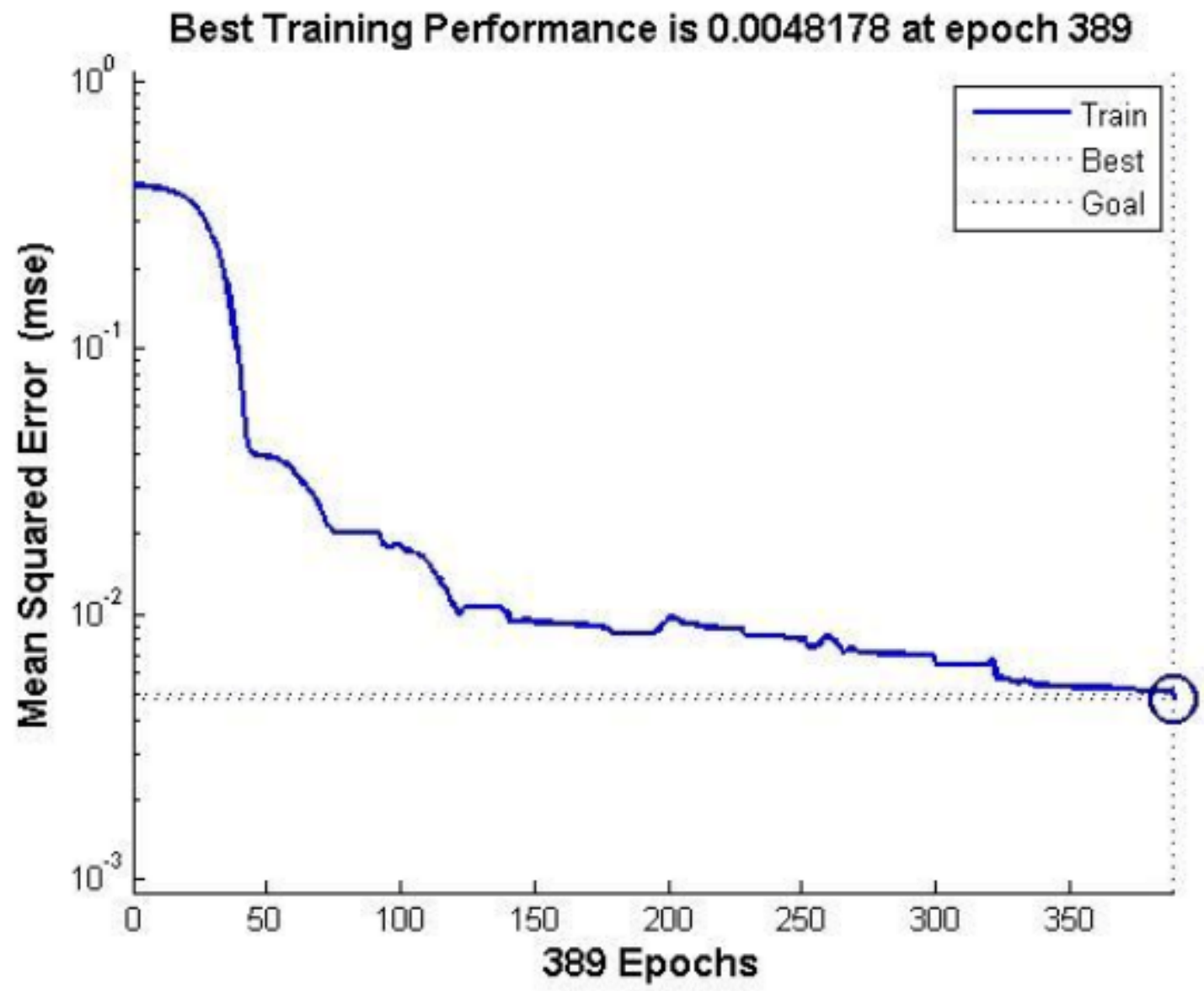


图 4.6 训练曲线

4.3.5 网络的输出

BP网络的设计目的是使其输出矢量在正确的位置上输出值为 1，而其他位置上输出为0。但是图像中的噪声输入矢量容易导致网络 1和0的输出值不正确，或输出其他错误值。为了增加网络的抗干扰能力， 在网络训练完成后， 再将其输出值经过一层竞争网络的处理，筛选出最接近输入值的输出位置，将输出置为 1，其他位置为 0。0-9数字的期望值与输入 1时的输出值如表 4.1所示。经竞争网络输出为 1符合要求。

表 4.1　输入 0~9 数字网络的期望输出表

输入数 输　　字 出神经元	0	1	2	3	4	5	6	7	8	9
神经元 0	1	0	0	0	0	0	0	0	0	0
神经元 1	0	1	0	0	0	0	0	0	0	0
神经元 2	0	0	1	0	0	0	0	0	0	0
神经元 3	0	0	0	1	0	0	0	0	0	0
神经元 4	0	0	0	0	1	0	0	0	0	0
神经元 5	0	0	0	0	0	1	0	0	0	0
神经元 6	0	0	0	0	0	0	1	0	0	0
神经元 7	0	0	0	0	0	0	0	1	0	0
神经元 8	0	0	0	0	0	0	0	0	1	0
神经元 9	0	0	0	0	0	0	0	0	0	1

4.4　结果分析

表 4.2　神经网络字符识别结果

字符	正确次数	误码次数	字符	正确次数	误码次数	字符	正确次数	误码次数
苏	37	2	A	55	1	N	3	0
桂	20	0	B	6	1	Q	5	0
0	23	0	C	6	0	S	1	0
1	8	8	D	0	2	T	4	0
2	13	13	E	10	0	U	1	0
3	23	2	F	8	0	V	4	0
4	2	0	G	4	1	W	2	0
5	17	0	H	1	0	X	1	0
6	11	13	J	4	0	Y	3	0
7	22	1	K	3	1	Z	0	2
8	37	0	L	4	0			
9	24	2	M	9	0			

为了更直观的展现识别准确率问题，我将所有车牌每个车牌拆成　　7个字符，按字

符进行统计正确次数与误码次数结果如表　4.2所示。

由上面数据分析得出误码率为：　$\frac{49}{420}$　100%　11.7% 。其中汉字的误码率为：$\frac{2}{2\quad 57}$　100%　3.5%，大写英文字母的误码率为　$\frac{8}{8\quad 134}$　100%　5.6%，数字的误码率为　$\frac{39}{39\quad 180}$　100%　17.8% 。误码率明显降低了好多。　结果中误识别的字符主要有　A和6、2和V，数字的识别误码率也高于字符，原因可能是数字相比字符复杂度更高。

造成识别字符出错的原因可能有：　字符切割时边框去没有除净；　图像有噪声，　或者识别参数设置科学根据不强，引起误差。

4.5　本章小结

本章节详细介绍了本设计的　BP神经网络识别字符的网络结构与设计，各参数的设定，以及特征选取。通过训练网络使网络具备字符识别的功能，　由最后的结果分析可以看出，神经网络字符识别具有相当高的准确率，值得进一步研究改进。

5 GUI 运行界面设计

MATLAB 软件提供了两种设计图形界面的方法：低层句柄图形对象命令和使用 GUI 开发环境的图形界面。其中 GUI 开发环境的设计方式更加简洁方便。 GUI 开发环境提供了一个包括工具栏、菜单、控件面板和空白界面的设计区。

5.1 图形用户界面

在 MATLAB 的图形用户界面中，包含了文本框、按钮等一系列交互控件，使用户可以直接用鼠标和键盘进行交互式操作。如图 5.1 所示。

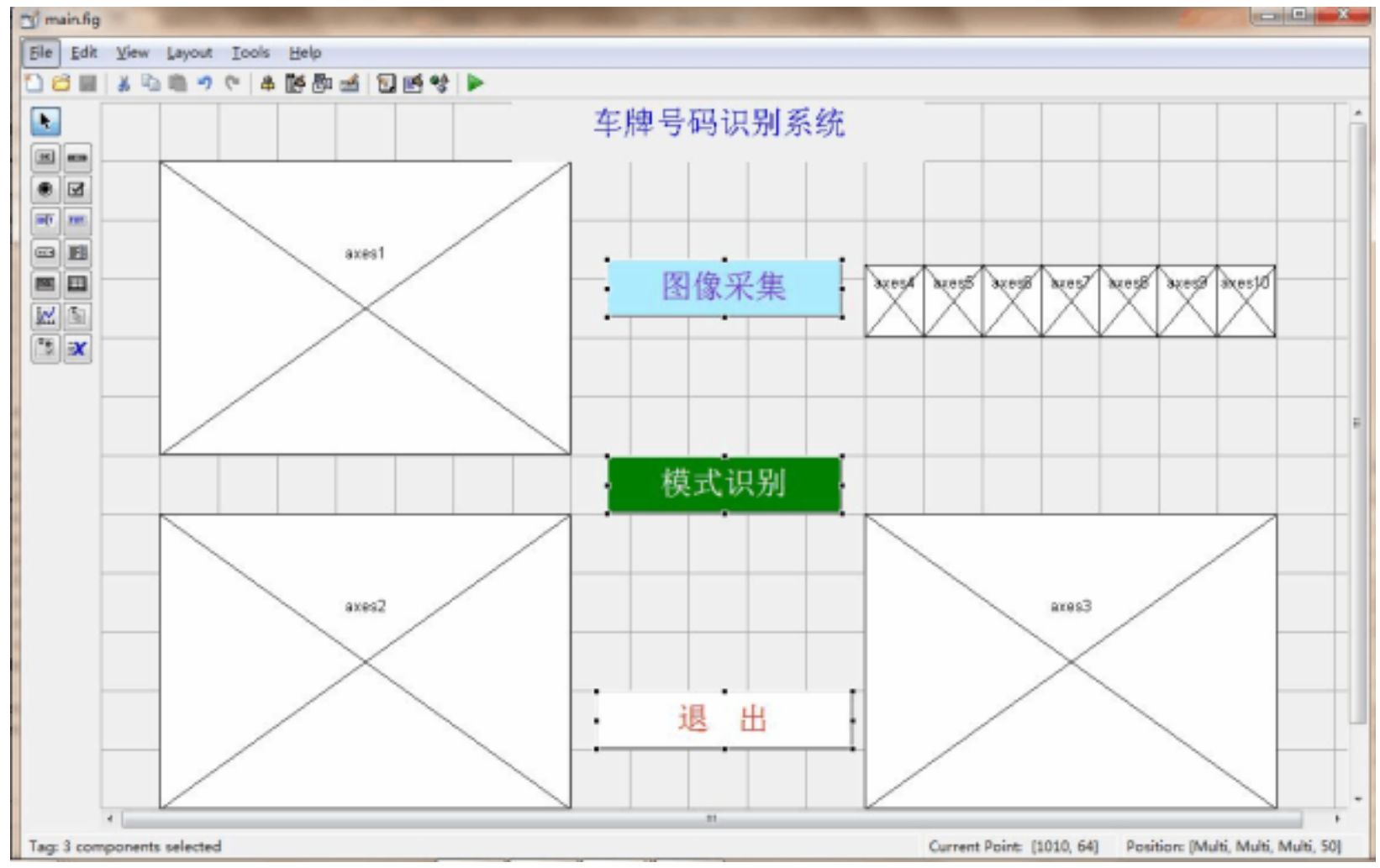


图 5.1 空白的图形界面窗口

左侧为控件面板，中间网格区域为界面设计区域，上方工具条有：运行、对象浏览器、属性编辑器、 M 文件编辑器、 Tab 顺序编辑器、 菜单编辑器、 对象对其工具等。

和 VC++ 、Java 等支持图形用户界面的软件类似， 在 GUI 开发环境中设计图形用户界面也是通过简单的鼠标拖拽来实现前台界面的设计。 后台的程序设计则是通过编写交互控件的回调函数。 MATLAB 很好的将实现程序功能的内核代码与交互控件的事件很好的关联起来。

5.2 GUI 设计界面的操作步骤

为了突出 MATLAB 软件的直观性，将本系统做成了可直接鼠标操作的 GUI 界面。以下为该系统的运行步骤：

（1）打开 MATLAB 软件，添加车牌号码识别系统的所有文件。

（2）运行 main.m 文件或者接打开系统的 GUI 界面 main.fig 文件。

（3）main.fig 界面中有四个按钮， 分别是图像采集、模式识别、神经网络识别和退出。点击图像采集，在文件夹中选择要识别的图片， axes1 中显示原图，接着就可以点击模式识别或者神经网络识别按钮， 系统就会自动采用模式识别或者神经网络算法识别图片，处理过程会相应的在界面上显示。

（4）识别输出结果后可以选择继续识别其他图片或点击退出。如图 5.2 示。

图 5.2 GUI 界面

结束语

通过近几个月的坚持学习，查阅了大量的资料。我从起初的只会 MATLAB 的简单操作，再到读懂程序，再到能自己完成程序的编程与修改，函数的调用，对算法的学习，最终成功完成了本次的车牌号码识别系统的制作。

在设计中我遇到了很多问题： 如在车牌定位时对于一些颜色混杂的图像， 车牌总是定位不准，后来通过查阅资料发现单纯的基于颜色特征的车牌定位法极易受到图像颜色复杂的影响， 于是我将定位改进成基于边缘检测和纹理特征的定位方法， 效果改善了很多。起初在字符识别研究中，模式识别的误码率极高，通过和导师的交流，导师建议我尝试比较标准的模板进行尝试， 先后用了 3 组模板进行试验， 发现清晰标准的模板确实准确率略高一些。 在指导老师的耐心提点下， 我少走了很多弯路， 从中也学习到了很多研究的思想与方法， 我会好好总结这次毕业设计中的经验， 将其投入到我今后的工作生活之中。

本次设计的系统还是存在了一些不足和局限性， 毕竟在投入生产使用还有很长的路要走，但我尽自己所能的追求识别准确率的最大化。 科学的进步过程就是要不断追求更优的过程， 在这个经济高速发展的国家， 车牌识别研究价值很高， 期待我国的车牌识别系统研究更上一层楼。

致　谢

四年的大学生活即将以这次设计画上一个句号，　我认真的投入了这次设计，　也为我的大学生活添上了浓墨重彩的一笔。　在这一刻，我将迎来心得人生转折，　漫漫求学路，我要感谢养育我的家人，教育我的师长，陪伴我的同学，默默关注我，帮助我的学长和学姐们，以及那些给我挫折的竞争对手们，没有你们我的路途不会如此充实。

在此论文即将付梓之际，　我的心情无比激动。　我要将我最大的敬意送给我的导师温老师。我并不是成绩最好的学生，　但您却是我最尊敬的老师。　您的学识渊博，　待人和蔼，令我能够毫无避讳的像您请教，　让我学的更多。　可以说我这次的设计是站在了巨人的肩膀上，温老师是我坚强的后盾，是我探索知识的风向标，在您的引导下，我才能顺利完成这次设计，也让我切身体会到，学海无涯，学无止境的道理。

我还要感谢我的家人，　使他们默默的支持才让我无需被生活的窘迫所扰，　专心学习知识，祝愿你们永远健康快乐。从资料的收集，到最后的设计完成，有多少可敬的师长，同学，以及论坛的网友，讨论群里的朋友给我提供了帮助，在这里清接受我诚挚的谢意。同时也要感谢学校图书馆为我提供了良好的学习资源。　最后再一次感谢为我提供帮助给我前进动力的人，和在设计中被我引用或参考的论著的作者们。

参 考 文 献

[1] 张国伍 . 智能交通系统工程导论 [M]. 北京：电子工业出版社, 2003.

[2] 侯海滨，沈希忠 . 车辆牌照识别技术的研究发展 [J]. 上海应用技术学院学报, 2009, (3) ：9～15.

[3] 车牌识别系统的研究背景意义及国内外研究现状 [DB/OL] http://wenku.baidu.com/view/9559f604de80d4d8d15a4f0a.html.

[4] 张学工 . 模式识别 (第 3 版)[M]. 北京：清华大学出版社, 2010.

[5] 何书前，张学平 . 车牌识别关键技术的应用研究 [J]. 电脑知识与技术学报, 2009, (9) ：5～12.

[6] 王敏，黄心汉，魏武，李炜 . 一种模板匹配和神经网络的车牌字符识别方法 [J]. 华中科技大学学报, 2001,（3）：10～12.

[7] 陈杰 .MATLAB宝典 [M]. 北京：电子工业出版社, 2010.

[8] 石红兰 . 基于图像处理的车牌识别系统的研究与实现 [J]. 机电信息学报, 2011, (21) ：15～22.

[9] 王雪 . 车牌图像预处理及识别算法研究 [J]. 机电产品开发与创新, 2010, (6) ：9～12.

[10] 刘聪 . 车牌识别系统关键技术的研究与实现 [D]. 西安：西北大学, 2012.

[11] 陆兴娟，吴震宇 . 图像边缘检测算法研究 [J]. 现代电子技术, 2010, (6) ：13 ～15.

[12] 白利波 . 车牌检测与识别算法研究 [M] , 北京交通大学出版社, 2007.

[13] 李盛宁 . 车牌识别系统中车牌定位算法的研究 [D]. 苏州：苏州大学, 2011.

[14] 陈虹 . 汽车车牌的自动检测与识别 [J]. 交通建设与管理, 2009, (11) ：4 ～12.

[15] Ostu N. A Threshold selection method from gray-level histograms[J]. IEEE Trans Systems , Man and Cybernetics , 1979, 9(1). 62 ～66.

[16] 陈丹 . 一种改进的文本图像二值化算法 . 计算机工程 . 2003 , 29(13). 1 ～3.

[17] 钱成 . 车牌识别中字符分割的研究 [J]. 中国科技论文在线, 2011, (1) ：19 ～22.

[18] 张引，潘云鹤 . 面向车辆牌照字符识别的预处理算法 . 计算机应用研究 . 1999 , 3(7). 5 ～16.

[19] 边肇祺，张学工 . 模式识别 [M]. 北京：清华大学出版社, 1988.

[20] 焦李成 . 神经网络系统理论 [M]. 西安：西安电子科技大学出版社, 1996.

[21] 黄德双 . 神经网络模式识别系统理论 [M]. 北京：电子工业出版社, 1996.

附录 A 模式识别主程序代码

```
function varargout = main(varargin)

% 开始初始化代码 - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @main_OpeningFcn, ...
                   'gui_OutputFcn',  @main_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% 结束初始化代码
pause(1);
% 执行之前主要是可见的。
function main_OpeningFcn(hObject, eventdata, handles, varargin)
set(handles.process,'enable','on')

% 此函数没有输出参数, 见 OutputFcn.
% main 的选择默认的命令行输出
handles.output = hObject;
% 更新 handles 结构
guidata(hObject, handles);
% UIWAIT 等待用户响应 (see UIRESUME)
% uiwait(handles.figure1);

% --- 这个函数的输出返回到命令行。
function varargout = main_OutputFcn(hObject, eventdata, handles)
tic
% 获得缺省命令行输出的把手结构
varargout{1} = handles.output;

% --- 执行按钮在 pushbutton1。
```

```
function pushbutton1_Callback(hObject, eventdata, handles)

[filename pathname]=uigetfile({'*.jpg';'*.bmp'}, 'File Selector');
I = imread([pathname '\' filename]);
handles.I = I;
%  更新处理结构
guidata(hObject, handles);
axes(handles.axes1);
imshow(I);title('  原始图片  ')

set(handles.process,'enable','on')
% ---   执行过程中按下按钮。

function process_Callback(hObject, eventdata, handles)
I = handles.I;
I1=rgb2gray(I);;%rgb2gray   转换成灰度图
guidata(hObject, handles);
axes(handles.axes2);
imshow(I1);title('  灰度图  ');
axes(handles.axes3);
imhist(I1);title('  灰度图直方图   ');
%继续
pause(2);
I2=edge(I1,'robert',0.15,'both');
guidata(hObject, handles);
axes(handles.axes2);
imshow(I2);title('robert   算子边缘检测   ');
pause(2);
se=[1;1;1];
I3=imerode(I2,se);
guidata(hObject, handles);
axes(handles.axes3);
imshow(I3);title('  腐蚀后图像  ');
%继续
pause(2);
se=strel('rectangle',[10,25]);% 生成一个矩阵
I4=imclose(I3,se);% 闭运算
guidata(hObject, handles);
axes(handles.axes2);
imshow(I4);title('  平滑图像的轮廓   ');
%继续
pause(2);
I5=bwareaopen(I4,2000);% 小于 2000的对象都被删除
guidata(hObject, handles);
```

```
axes(handles.axes2);
imshow(I5);title('  从对象中移除小对象    ');
%继续
pause(2);
[PY2,PY1,PX2,PX1]=chepai_fenge(I5);%   调用分割车牌
global threshold;
[PY2,PY1,PX2,PX1,threshold]=chepai_xiuzheng(PY2,PY1,PX2,PX1);%     调用车牌校正
IY=I(PY1:PY2,:,:);
Plate=I5(PY1:PY2,PX1:PX2);%  使用 caitu_tiqu
  global dw;
  dw=Plate;
PX1=PX1-1;% 对车牌区域的校正
  PX2=PX2+1;
    dw=I(PY1:PY2-8,PX1:PX2,:);
    axes(handles.axes2);
    imshow(dw),title('  车牌区域的校正   ');
  pause(2);
  t=tic;
guidata(hObject, handles);
axes(handles.axes2);
imshow(IY),title('  水平方向合理区域    ');
axes(handles.axes3);
imshow(dw),title('  定位剪切后的彩色车牌图像     ');
pause(2);
imwrite(dw,'New number plate.jpg');
[filename,filepath]=uigetfile('New number plate.jpg','    输入一个定位裁剪后的车牌图像      ');
jpg=strcat(filepath,filename);
a=imread('New number plate.jpg');
b=rgb2gray(a);% 对定位后的车牌灰度化
figure(3),subplot(3,2,1),imshow(b),title('   车牌灰度图像   ');
g_max=double(max(max(b)));
g_min=double(min(min(b)));
T=round(g_max-(g_max-g_min)/2); %T    为二值化的阈值
[m,n]=size(b);
d=(double(b)>=T);     %    d:二值图像
figure(3),subplot(3,2,2),imshow(d),title('   车牌二值图像   ');
figure(3),subplot(3,2,3),imshow(d),title('   均值滤波前   ');
pause(1);
h=fspecial('average',3); % 均值滤波器
d=im2bw(round(filter2(h,d)));
figure(3),subplot(3,2,4),imshow(d),title('   均值滤波后   ');
se=eye(2); % eye(n) returns the n-by-n identity matrix     单位矩阵
%字符面积与车牌面积之比在     (0.235,0.365) 之间
[m,n]=size(d);     %如果大于  0.365则对图像进行腐蚀，如果小于     0.235则对图像进行膨胀
```

```
if bwarea(d)/m/n>=0.365%  计算面积
        d=imerode(d,se);%imerode   实现图像腐蚀     d为待处理图像,      se是结构元素对象
elseif bwarea(d)/m/n<=0.235
        d=imdilate(d,se);%imdilate     图像膨胀
end
figure(3),subplot(3,2,5),imshow(d),title('    膨胀或腐蚀处理后    ');
pause(2);
%  寻找连续有文字的块，若长度大于某阈值，则认为该块有两个字符组成，需要分割
d=zifufenge(d);
[m,n]=size(d);
guidata(hObject, handles);
axes(handles.axes3);imshow(d);
k1=1;k2=1;s=sum(d);j=1;
while j~=n
        while s(j)==0
                j=j+1;
        end
        k1=j;
        while s(j)~=0 && j<=n-1
                j=j+1;
        end
        k2=j-1;
        if k2-k1>=round(n/6.5)
                [val,num]=min(sum(d(:,[k1+5:k2-5])));
                d(:,k1+num+5)=0     %  分割
        end
end
%  再分割
d=zifufenge(d);
%  切割出    7  个字符
y1=10;y2=0.25;flag=0;word1=[];
while flag==0
        [m,n]=size(d);
        left=1;wide=0;
        while sum(d(:,wide+1))~=0
                wide=wide+1;
        end
        if wide<y1       %    认为是左侧干扰
                d(:,[1:wide])=0;
        d=zifufenge(d);
else
        temp=zifufenge(imcrop(d,[1 1 wide m]));
        [m,n]=size(temp);
        all=sum(sum(temp));
```

```
            two_thirds=sum(sum(temp([round(m/3):2*round(m/3)],:)));
            if two_thirds/all>y2
                flag=1;word1=temp;
            end
            d(:,[1:wide])=0;
            d=zifufenge(d);
        end
end
%  分割出第二个字符
[word2,d]=getword(d);
pause(1);
%  分割出第三个字符
[word3,d]=getword(d);
pause(1);
%  分割出第四个字符
[word4,d]=getword(d);
pause(1);
%  分割出第五个字符
[word5,d]=getword(d);
pause(1);
%  分割出第六个字符
[word6,d]=getword(d);
pause(1);
%  分割出第七个字符
[word7,d]=getword(d);
pause(1);
guidata(hObject, handles);
axes(handles.axes4);imshow(word1),title('1');
guidata(hObject, handles);
axes(handles.axes4);imshow(word2),title('2');
guidata(hObject, handles);
axes(handles.axes4);imshow(word3),title('3');
guidata(hObject, handles);
axes(handles.axes4);imshow(word4),title('4');
guidata(hObject, handles);
axes(handles.axes4);imshow(word5),title('5');
guidata(hObject, handles);
axes(handles.axes4);imshow(word6),title('6');
guidata(hObject, handles);
axes(handles.axes4);imshow(word7),title('7');
[m,n]=size(word1);
%  商用系统程序中归一化大小为        40*20, 此处演示
word1=imresize(word1,[40 20]);
word2=imresize(word2,[40 20]);
```

```
word3=imresize(word3,[40 20]);
word4=imresize(word4,[40 20]);
word5=imresize(word5,[40 20]);
word6=imresize(word6,[40 20]);
word7=imresize(word7,[40 20]);
guidata(hObject, handles);
axes(handles.axes4);imshow(word1),title('1');
guidata(hObject, handles);
axes(handles.axes5);imshow(word2),title('2');
guidata(hObject, handles);
axes(handles.axes6);imshow(word3),title('3');
guidata(hObject, handles);
axes(handles.axes7);imshow(word4),title('4');
guidata(hObject, handles);
axes(handles.axes8);imshow(word5),title('5');
guidata(hObject, handles);
axes(handles.axes9);imshow(word6),title('6');
guidata(hObject, handles);
axes(handles.axes10);imshow(word7),title('7');
imwrite(word1,'1.jpg');
imwrite(word2,'2.jpg');
imwrite(word3,'3.jpg');
imwrite(word4,'4.jpg');
imwrite(word5,'5.jpg');
imwrite(word6,'6.jpg');
imwrite(word7,'7.jpg');
liccode=char(['0':'9' 'A':'Z' '   桂鲁苏豫京  ']);    %建立自动识别字符代码表
SubBw2=zeros(40,20);    %初始值归为  0矩阵
l=1;
for l=1:7
          ii=int2str(l);%  将整数转换为字符串
        t=imread([ii,'.jpg']);
          SegBw2=imresize(t,[40 20],'nearest');%  实用最近邻插值法放大图像
             if l==1                                %第一位汉字识别
                    kmin=37;
                    kmax=41;
                pause(1);
        elseif l==2                                 %第二位   A~Z  字母识别
                kmin=11;
                kmax=36;
                pause(1);
        else l>=3                                   %第三位以后是字母或数字识别
                kmin=1;
                kmax=36;
```

```
end
for k2=kmin:kmax
    fname=strcat(' 模板库 \',liccode(k2),'.jpg');      %读取字符模板库中的图像
    SamBw2 = imread(fname);
    for     i=1:40
        for j=1:20
            SubBw2(i,j)=SegBw2(i,j)-SamBw2(i,j);
        end
    end
                % 以上相当于两幅图相减得到第三幅图
     Dmax=0;
     for k1=1:40
        for l1=1:20
            if     ( SubBw2(k1,l1) > 0 || SubBw2(k1,l1) <0 )
                Dmax=Dmax+1;
%如果两幅图像对应的图像相减得到不是       0,  次数加  1
            end
        end
    end
    Error(k2)=Dmax;
end
Error1=Error(kmin:kmax);
            MinError=min(Error1);
%如果第三幅图像所得到     0数量最多，那么两幅图像相似度最高
            findc=find(Error1==MinError);
            Code(l*2-1)=liccode(findc(1)+kmin-1);
            Code(l*2)=' ';%  确定识别出的图像在车牌中的位置，中间加上空格
            l=l+1;     %继续循环
end
guidata(hObject, handles);
axes(handles.axes3);
imshow(dw),title (['   车牌识别号码    :', Code],'Color','b');
pause(2);
fid = fopen('Data.xls', 'a+');
  fprintf(fid,'%s\r\n',Code,datestr(now));
  winopen('Data.xls');
  fclose(fid);
function pushbutton6_Callback(hObject, eventdata, handles)
close(gcf);


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over text5.
function text5_ButtonDownFcn(hObject, eventdata, handles)
```

% hObject handle to text5 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDA TA)

主程序所调用车牌分割子程序如下：

```
function [PY2,PY1,PX2,PX1]=chepai_fenge(I5)
[y,x,z]=size(I5);
myI=double(I5);
%begin横向扫描
tic
Y_threshlow=5; %这个数值很重要。决定了提取的彩图的质量
X_firrectify=5;
%=== Y 方向=== 从左向右寻找第一个 1值像素大于 5的坐标为水平方向左侧分界线, 从优向左寻找到第一个 1值像素量大于 5的为右侧分界线
Blue_y=zeros(y,1);
for i=1:y
    for j=1:x
        if(myI(i,j,1)==1)% 如果myI(i,j,1) 即myI图像中坐标为 (i,j) 的点为白色
         %则Blue_y的相应行的元素 white_y(i,1)值加 1
            Blue_y(i,1)= Blue_y(i,1)+1;          % 蓝色象素点统计
        end
    end
end
        [temp MaxY]=max(Blue_y);
    % Y方向车牌区域确定 temp(最多点数 ):所有行中, 最多的累积像素点 MaxY（最多点所在行） :该行中蓝点最多
     PY1=MaxY;% 有最多蓝点的行付给 PY1
    while ((Blue_y(PY1,1)>=Y_threshlow)&&(PY1>1))% 找到图片上边界
        PY1=PY1-1;
    end

    %PY1：存储车牌上边界值
    PY2=MaxY;
    while ((Blue_y(PY2,1)>=Y_threshlow)&&(PY2<y))% 阈值为 5
        PY2=PY2+1;
    end
    PY1, PY2 %原始图像 I中截取的纵坐标在 PY1： PY2之间的部分
    figure(1),imshow(Blue_y),title('y 方向确定 ');
    pause(2);
    %==============X 方向 ==================================
    X_threshhigh=(PY2-PY1)/11;
%这个数值很重要。 决定了提取的彩图的质量 ,适当提高可抗干扰, 但是小图会照成剪裁太多
    Blue_x=zeros(1,x);                  % 进一步确定 X方向的车牌区域
    for j=1:x
```

```
        for i=PY1:PY2
            if(myI(i,j,1)==1)
              Blue_x(1,j)= Blue_x(1,j)+1;
            end
        end
end
[temp MaxX]=max(Blue_x);
PX1=MaxX-6*(PY2-PY1);
if    PX1<=1
       PX1=1;
end
while ((Blue_x(1,PX1)<=X_threshhigh)&&(PX1<x))%  阈值
          PX1=PX1+1;
end                          %确定出 X方向车牌起点
PX2=MaxX+6*(PY2-PY1);
if    PX2>=x
       PX2=x;
end
while ((Blue_x(1,PX2)<=X_threshhigh)&&(PX2>PX1))%  阈值
           PX2=PX2-1;
end                    %确定出 X方向车牌终点
PX1 ,PX2
                figure(2),imshow(Blue_x),title('X 方向确定 ');
```

主程序所调用车牌修正子程序如下：

```
function [PY2,PY1,PX2,PX1,threshold]=chepai_xiuzheng(PY2,PY1,PX2,PX1) %      车牌修正
S=(PY2-PY1)*(PX2-PX1)
if              S<=25000     %25000是像素
        threshold=50;
        Y_secrectify=3;
        X_secrectify=3;
elseif S>25000&&S<=45000
        threshold=100;
        Y_secrectify=-3;
        X_secrectify=3;
elseif S>45000&&S<=80000
        threshold=200;
        Y_secrectify=-3;
        X_secrectify=3;
elseif S>80000&&S<=150000
        threshold=300;
        Y_secrectify=-10;
        X_secrectify=-10;
elseif S>150000&&S<=400000
        threshold=600;
```

```
        Y_secrectify=-10;
        X_secrectify=-10;
else
        threshold=1800;
        Y_secrectify=-10;
        X_secrectify=-10;
end
PY1=PY1-Y_secrectify;
PY2=PY2+Y_secrectify;
        PX1=PX1;
        PX2=PX2;% 对车牌区域的进一步修正
```

主程序所调用字符分割子程序如下：

```
        function e=zifufenge(d)        %字符分割
        [m,n]=size(d);
        top=1;bottom=m;      %m 为高
        left=1;right=n;      %n为宽
        while sum(d(top,:))==0 && top<=m          %切割出白色区域（横切）
                top=top+1;
        end
        while sum(d(bottom,:))==0 && bottom>=1         %同上
                bottom=bottom-1;
        end
        while sum(d(:,left))==0 && left<=n                %切割出白区域（纵切）
                left=left+1;
        end
        while sum(d(:,right))==0 && right>=1
                right=right-1;
        end
        dd=right-left;
        hh=bottom-top;
        e=imcrop(d,[left top dd hh]);        %在一个数字窗口显示图像      d
```

主程序所调用字符识别子程序如下：

```
        function [word,result]=getword(d)%    获取字符
        word=[];flag=0;y1=8;y2=0.5;
                while flag==0
                [m,n]=size(d);
                wide=0;
                while sum(d(:,wide+1))~=0 && wide<=n-2
%有白色加  1知道没有白色，也就是找出一个白色区域
                        wide=wide+1;
                end
                temp=zifufenge(imcrop(d,[1 1 wide m]));
                [m1,n1]=size(temp);
```

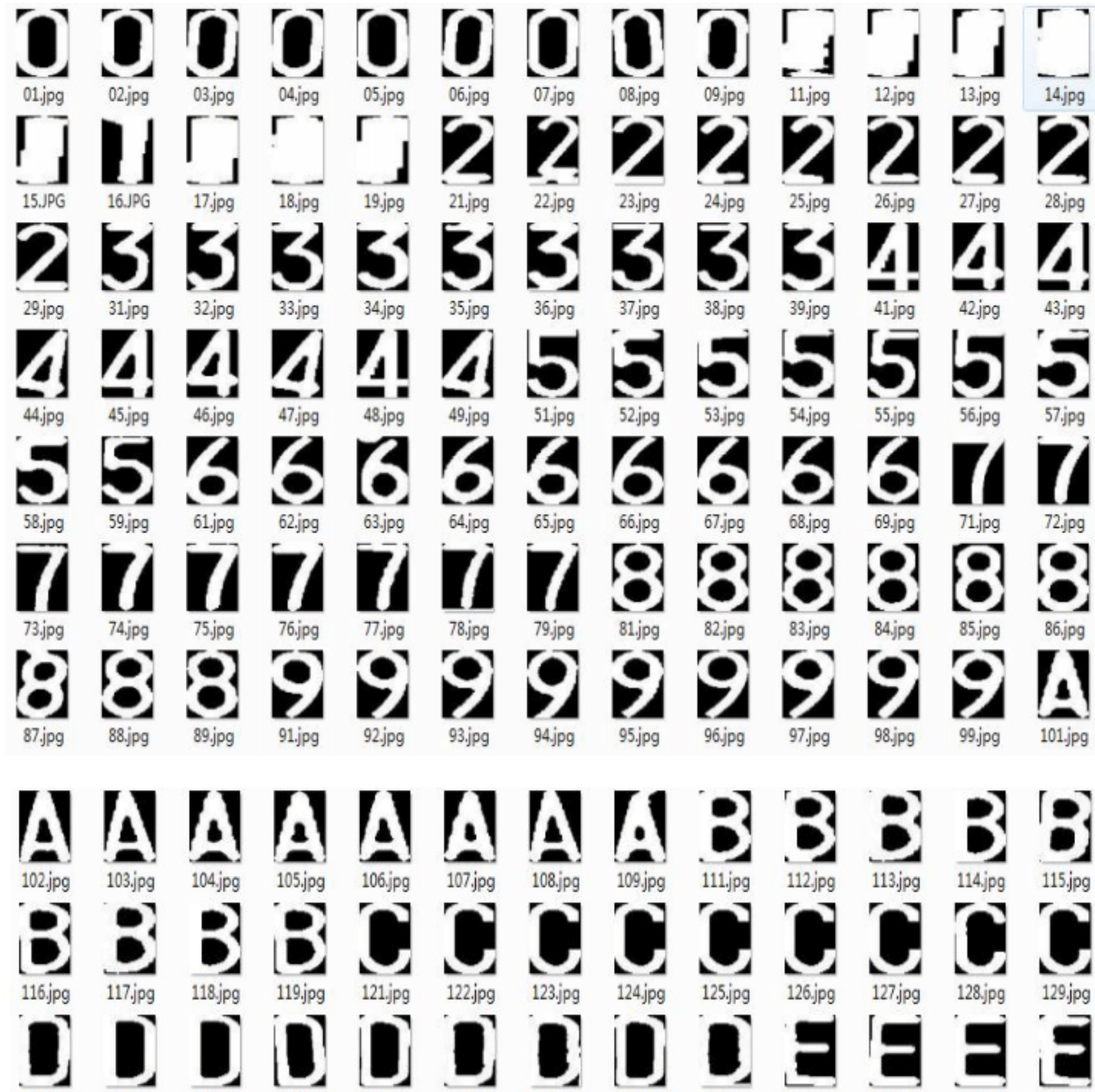
```
if wide<y1 && n1/m1>y2
        d(:,[1:wide])=0;
        if sum(sum(d))~=0
                d=zifufenge(d);     %  切割出最小范围
        else word=[];flag=1;
        end
else
        word=zifufenge(imcrop(d,[1 1 wide m]));
        d(:,[1:wide])=0;
                if sum(sum(d))~=0;
                        d=zifufenge(d);flag=1;
                else d=[];
                end
        end
end
           result=d;
```

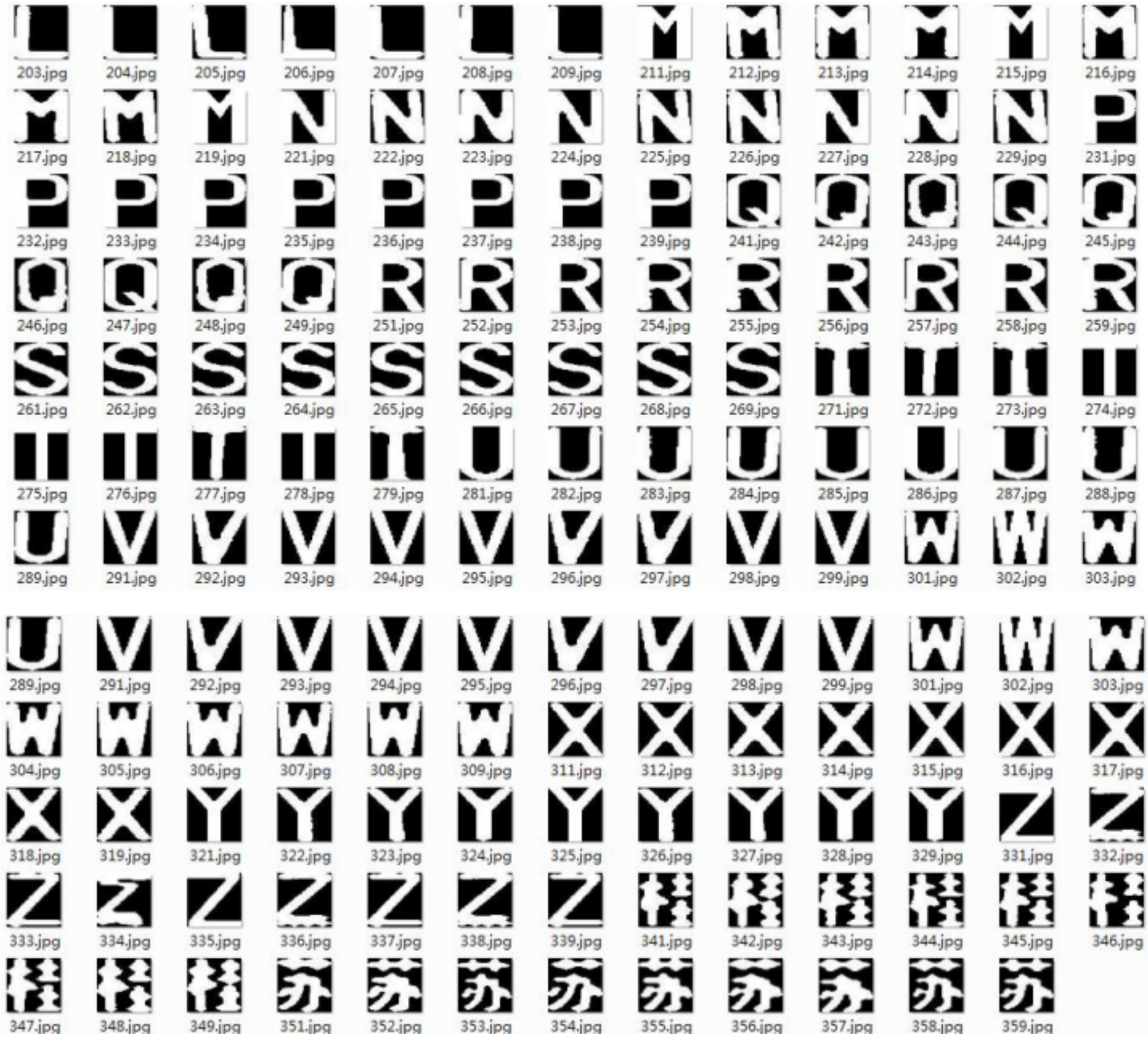
附录 C　图像模板

模式识别中的模板

用于研究的车牌图像

用于神经网络训练的字符主样本




203.jpg
204.jpg
205.jpg
206.jpg
207.jpg
208.jpg
209.jpg
211.jpg
212.jpg
213.jpg
214.jpg
215.jpg
216.jpg
217.jpg
218.jpg
219.jpg
221.jpg
222.jpg
223.jpg
224.jpg
225.jpg
226.jpg
227.jpg
228.jpg
229.jpg
231.jpg
232.jpg
233.jpg
234.jpg
235.jpg
236.jpg
237.jpg
238.jpg
239.jpg
241.jpg
242.jpg
243.jpg
244.jpg
245.jpg
246.jpg
247.jpg
248.jpg
249.jpg
251.jpg
252.jpg
253.jpg
254.jpg
255.jpg
256.jpg
257.jpg
258.jpg
259.jpg
261.jpg
262.jpg
263.jpg
264.jpg
265.jpg
266.jpg
267.jpg
268.jpg
269.jpg
271.jpg
272.jpg
273.jpg
274.jpg
275.jpg
276.jpg
277.jpg
278.jpg
279.jpg
281.jpg
282.jpg
283.jpg
284.jpg
285.jpg
286.jpg
287.jpg
288.jpg
289.jpg
291.jpg
292.jpg
293.jpg
294.jpg
295.jpg
296.jpg
297.jpg
298.jpg
299.jpg
301.jpg
302.jpg
303.jpg
289.jpg
291.jpg
292.jpg
293.jpg
294.jpg
295.jpg
296.jpg
297.jpg
298.jpg
299.jpg
301.jpg
302.jpg
303.jpg
304.jpg
305.jpg
306.jpg
307.jpg
308.jpg
309.jpg
311.jpg
312.jpg
313.jpg
314.jpg
315.jpg
316.jpg
317.jpg
318.jpg
319.jpg
321.jpg
322.jpg
323.jpg
324.jpg
325.jpg
326.jpg
327.jpg
328.jpg
329.jpg
331.jpg
332.jpg
333.jpg
334.jpg
335.jpg
336.jpg
337.jpg
338.jpg
339.jpg
341.jpg
342.jpg
343.jpg
344.jpg
345.jpg
346.jpg
347.jpg
348.jpg
349.jpg
351.jpg
352.jpg
353.jpg
354.jpg
355.jpg
356.jpg
357.jpg
358.jpg
359.jpg