苏州大学

数字图像处理

课程设计报告

（2016 年度春季学期）

项目名称：车牌图像识别系统

2016 年 12 月 31 日

大纲

一、小组概况

小组成员、分工等…

二、问题概要

问题描述及技术难点分析。

三、系统架构

系统流程图及技术平台。

四、系统实现

系统模块介绍及说明文档。

五、总结

系统实现成果总结。

附件：

1）附图

2）代码页

一、小组概况

小组成员	沈家赟
	吴晓灿
	戴星辰
	陆佳程
	高冰
	张珍茹
	魏然
	封琳
	丁瑞宇
	沈维铃
专业	2014 级 计算机科学与技术
学院	计算机科学与技术学院
任课老师	刘纯平 教授
报告版本	第 3 版
修订日期	2016 年 12 月 31 日

成员分工：

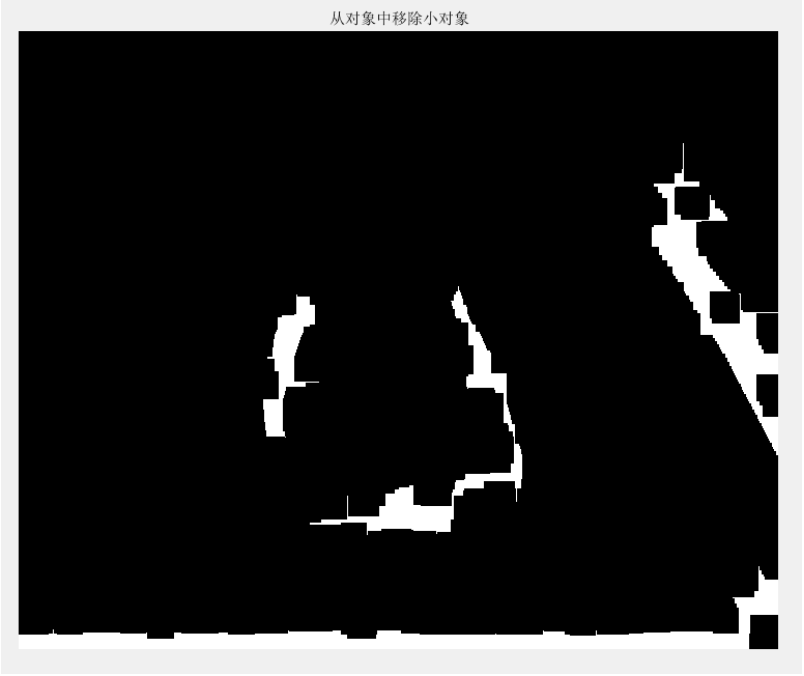
小组成员	分工
沈家赟	组长，代码原型，文档撰写、可执行文件生成
魏然	副组长，文字识别（recognizeWords）
张珍茹	文字识别（recognizeWords）
吴晓灿	原始图像预处理（preProcRawImg）
封琳	辅助函数（getword）
沈维铃	GUI 设计及编程（Matlab GUIDE）
戴星辰	定位裁剪车牌图像（getPlateImg），辅助函数（getBestArea）
高冰	车牌文字裁切（partitionWords）
陆佳程	车牌图像预处理（preProcPlateImg）
丁瑞宇	辅助函数（minimizeDistrict）

二、问题概要

车牌识别系统的目的是实现照片中车牌区域的特征提取和文字识别。基本架构大致包含预处理、图像分割、特征提取、识别与分类。

技术难点：

1） 原始图像预处理：遇到背景信息丰富的图片时，难以实现车牌提取：



原因在于远视角时，车牌在图像中比重较低，而其它无关信息较多，如：路沿、花坛、标志线、路灯等等。车辆自身多种颜色和反光特性也给原始图像预处理以及特征提取带来了很大的麻烦。如何有效地完成预处理，过滤杂讯，实现干净的特征提取时主要难点。

2）字符识别：

实现良好的字符切割效果并设计字符识别算法，实现字符图像到字符值的准确转换是字符识别的难点。

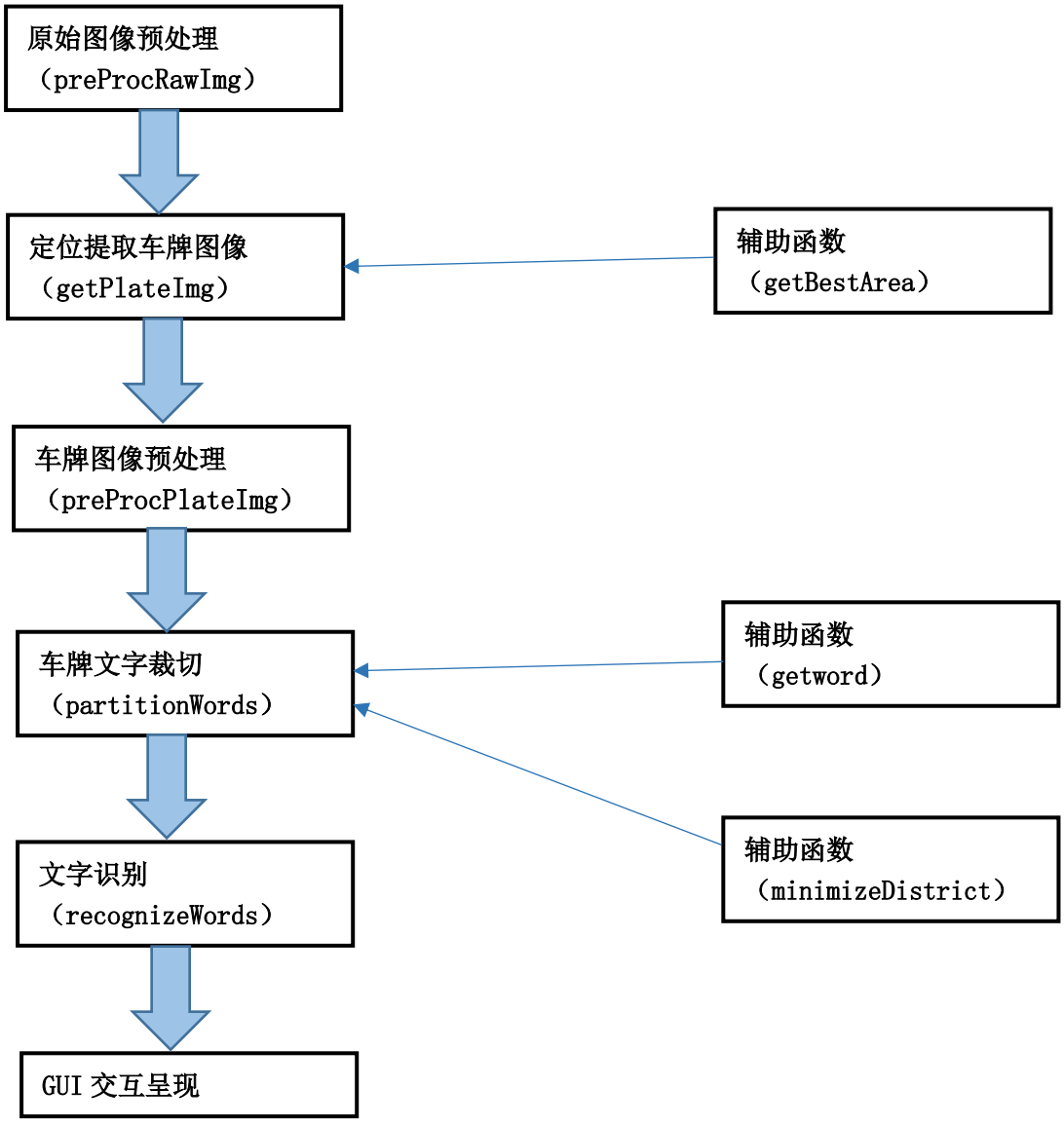
考虑到车牌采用法定统一标准，具有严格规定的格式和字体，因此，字符识别部分采用模板匹配是一个可靠合理且简单易行的方案。

但是，对于模板匹配实现字符识别对视角要求颇为严苛，要求诸多，如:视角不倾斜或偏移、上下或左右方向远近差异，无高反差光线，无显著尘垢等……模板匹配是一个简单易行的可行方案，应对实际复杂多变的使用场景仍有局限性。

采用神经网络机器学习的方式可能可以应对更复杂的图像场景。

三、系统架构

1、系统流程图



2、技术平台：Matlab

四、系统实现

主要函数模块：

1、 原始图像预处理 【技术难点】

I_Proced = preProcRawImg(I_raw)

负责人：吴晓灿

说明：

函数 preProcRawImg

输入 原图

输出 能够区分车牌和背景的二值图

原理

首先将原图像转化为灰度图，然后选定合适的阈值，将灰度图转化为二值图。用roberts 算子进行边缘检测，对边缘图像进行形态学腐蚀，之后进行合理地形态学闭合，最后将图像中的小对象移除，得到目标车牌所处的位置二值图。

2、定位裁剪车牌图像

I_plateRaw = getPlateImg(I_Proced, I_raw)

负责人：戴星辰

说明：

输入:预处理后的二值化图像与原图

输出：裁剪后的拆牌图像

图像定位与裁剪，即给定原图与对应的已与处理的二值化图像来裁剪出正确的车牌位置。

算法过程：

1.将预处理后的图像表明连通区域

2.合并相似的连通区域（某些时候车牌会被处理成不同的连通域）

3.查找长宽比最佳的连通域

4.对已查找到的连通域进一步细分四个边界（通过判断每一行列的非零像素点数）

5.根据边界从原图割出车牌区域。

3、车牌图像预处理

I_plateProced = preProcPlateImg(I_plateRaw)

负责人：陆佳程

说明：

输入为较为清楚的车牌周围的图像

输出为细化后的车牌二值图像

原理：

首先对输入图像进行二值化处理，然后利用 robert 算子边缘检测，检测出边缘后进行腐蚀然后平滑。处理后车牌部分是一个比较大的对象，把小对象移除后就能确定出车牌的坐标。利用坐标从原图中将车牌截出，然后就利用二值化、均值滤波、膨胀，就能得到预处理后的车牌图像。

4、车牌文字裁切

[word1,word2,word3,word4,word5,word6,word7] = partitionWords(I_plateProced)

负责人：高冰

说明：

输入输出：输入一个二值图像，输出对应的字符模板。

过程：

首先循环找出字符区域，判断字符区域大小，对于包含多个字符的区域，找出像素值最少的列，清空像素值，分开字符区域。

开始依次分割字符区域，并调用 getword 函数获取相对应的字符模板。

其中对于第一个字符区域，判断是否是字符，如果小于经验宽度，就不是字符，清空该区域像素值，进行切割。直至找到符合宽度的字符区域，并且该区域中间的像素值总和大于经验最小值，就认为是第一个字符，分割成功。

输出对应的字符模板。

5、文字识别 【技术难点】

Code = recognizeWords(word1, word2, word3, word4, word5, word6, word7)

负责人：张珍茹， 魏然

说明：

字符识别是车牌识别中很重要的一部分，在模式识别中也扮演的很重要的角色。这里使用的是模板匹配算法：

1、首先本文识别的前提是模板字符是二值图像，背景黑色，字符为白色；同样待识别的字符同样如此。

2、将预处理后的待识别字符图像 imageU 与字符模板库中的字符图像 imageT 进行”与“运算得到共同部分图像 imageV；

3、将得到的共同部分图像与待识别字符进行逻辑”异或“运算，得到待识别字符图像多余部分 imageX；

4、将得到的共同部分图像与模板字符进行逻辑”异或“运算，得到模板图像多余部分 imageW。

5、计算每个模板字符图像 imageT 中白像素个数 T,待识别字符图像 imageU 的白像素个数 U，

6、imageU 与 imageT 共同的部分 imageV 的白像素个数 V,imageW 的白像素个数 W；imageX 中白像素个数 X；

7、构造判别函数表达式为：

tempSum=sqrt(((T-TUV)*(T-TUV)+(U-TUV)*(U-TUV)+(V-TUV)*(V-TUV))/2);

Y(j)=V/(W/T*X/U*tempSum);

其中 TUV=（T+U+V）/3;

这样，相似系数最大 max（Y）对应的模板 M 为待识别字符；

辅助函数：

6、提取车牌文字区域

[word, result] = getword(d)

负责人：封琳

说明：

在汽车牌照自动识别过程中，字符分割有承前启后的作用。它在前期牌照定位的基础上进行字符的分割，然后再利用分割的结果进行字符识别。函数 function [word,result]=getword(d)用来对预处理后的车牌文字区域做提取，从车牌中分割出第 n 个字符。

（1） 开始的 d 为已经预处理过的车牌图像

（2） imcrop 是一个函数，在 MATLAB 中，该函数用于返回图像的一个裁剪区域。

[A,rect] = imcrop(...)格式指定了要裁剪的区域。imcrop(d,[1 1 wide m])为从坐标（1,1）处开始，裁剪出大小为（wide,m）的区域

（3） size() 函数：获取矩阵的行数和列数

[r,c]=size(A)当有两个输出参数时，size 函数将矩阵的行数返回到第一个输出变量 r，将矩阵的列数返回到第二个输出变量 c。

（4） sum(A; 列求和，以矩阵 A 的每一列为对象，对一列内的数字求和。

sum(A,2); 行求和，以矩阵 A 的每一行为对象，对一行内的数字求和。

sum(A(:)); 对矩阵进行全矩阵数值求和

（5） 如果切除的字符块 大于 y1,且宽高比为 1:2 ，就认为是有效字符 ，进行切割。

7、收缩、细化有效区域

I_out = minimizeDistrict(I_in)

负责人：丁瑞宇

说明：

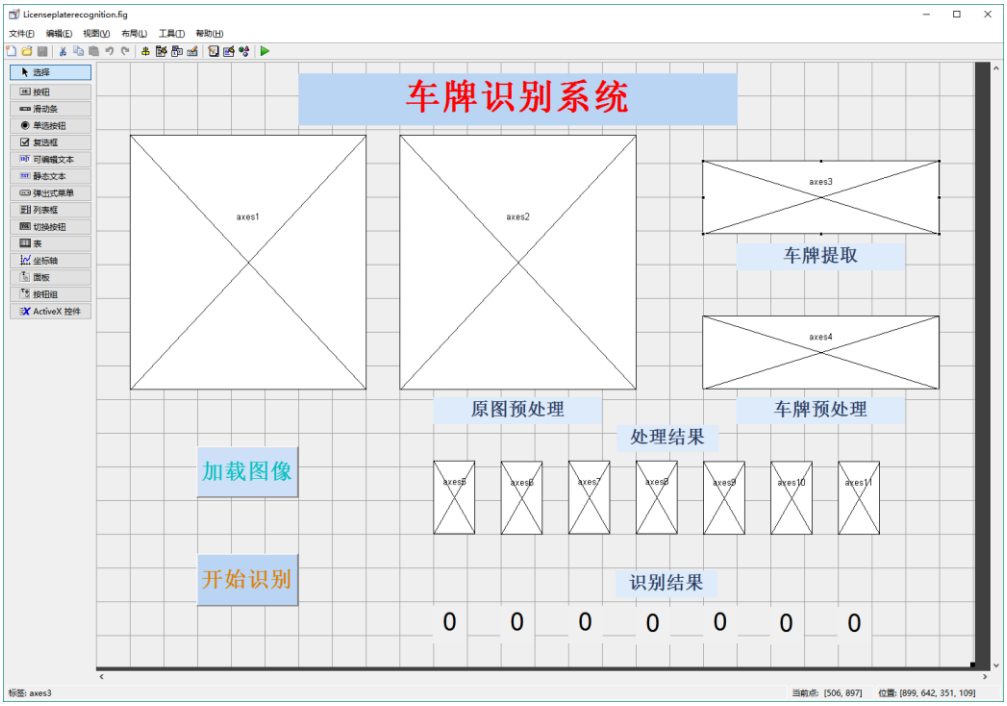
对预处理后的车牌文字区域再做收缩切割，即更新车牌区域。

GUI 交互：

8、GUI 设计及编程

负责人：沈维铃

GUIDE 设计布局：



Callback 等交互编程后，运行示意：

9、可执行文件生成

负责人：沈家赟

说明：

我编译了整个工程项目的 matlab 代码，根据 Matlab Documents 页，采用 mcc 命令可以编译 function 为可视化的可执行程序。

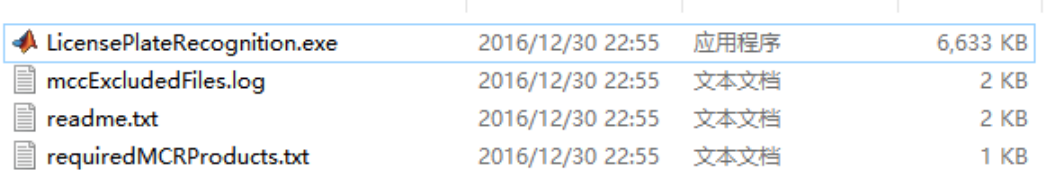
具体编译命令及属性配置如下：

```
mcc -e Licenseplaterecognition.m -a ./
```

注意：标准的 mcc -e 无法编译独立的依赖多函数及资源文件的程序，程序将无法脱离依赖函数及资源文件独立运行！必须配置 -a 属性，并提供工程文件夹路径，编译器才能递归包含所有依赖代码及资源文件。

编译得到的是 **Windows 平台，x64** 的可执行程序。经测试，编译得到的 EXE 格式可执行文件能够脱离依赖函数及资源文件（字符模板）独立运行。

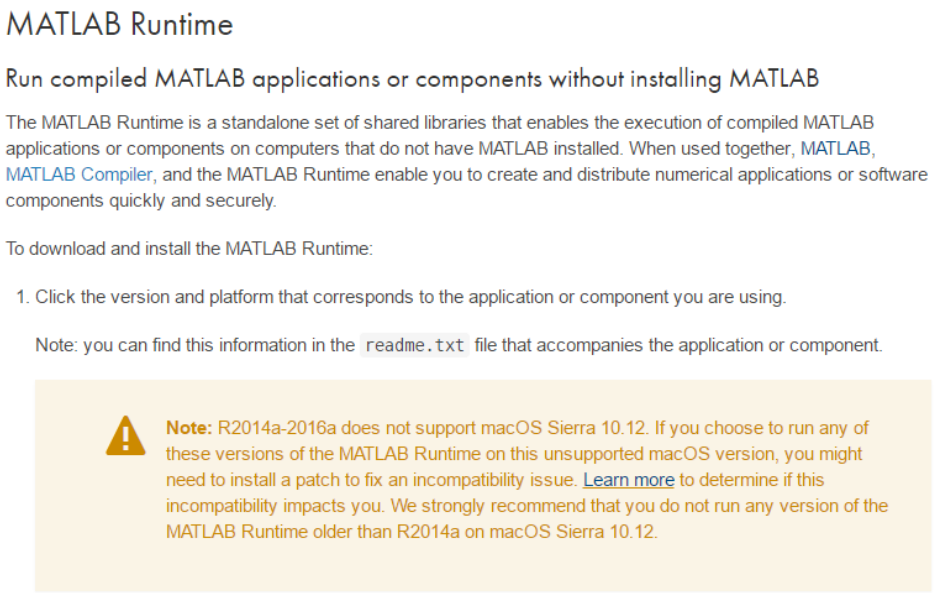
如需其它平台的可执行程序，须在对应平台的 Matlab 上编译。



LicensePlateRecognition.exe	2016/12/30 22:55	应用程序	6,633 KB
mccExcludedFiles.log	2016/12/30 22:55	文本文档	2 KB
readme.txt	2016/12/30 22:55	文本文档	2 KB
requiredMCRProducts.txt	2016/12/30 22:55	文本文档	1 KB

另注意：编译得到的 exe 可执行文件仅 6.47M，不包含 Matlab 运行时。如运行机未安装 Matlab，需要另行安装 Matlab 运行时环境 **MCR**（Matlab Runtime）。

详情请访问： http://www.mathworks.com/products/compiler/mcr/index.html



MATLAB Runtime

Run compiled MATLAB applications or components without installing MATLAB

The MATLAB Runtime is a standalone set of shared libraries that enables the execution of compiled MATLAB applications or components on computers that do not have MATLAB installed. When used together, MATLAB, MATLAB Compiler, and the MATLAB Runtime enable you to create and distribute numerical applications or software components quickly and securely.

To download and install the MATLAB Runtime:

1. Click the version and platform that corresponds to the application or component you are using.

Note: you can find this information in the `readme.txt` file that accompanies the application or component.

Note: R2014a-2016a does not support macOS Sierra 10.12. If you choose to run any of these versions of the MATLAB Runtime on this unsupported macOS version, you might need to install a patch to fix an incompatibility issue. Learn more to determine if this incompatibility impacts you. We strongly recommend that you do not run any version of the MATLAB Runtime older than R2014a on macOS Sierra 10.12.

此站点提供安装教程及各版本下载链接（建议安装 R2016a Runtime 或更新）

版本	Windows	Linux	Mac
R2016b (9.1)	64 位	64 位	Intel 64 位
R2016a (9.0.1)	64 位	64 位	Intel 64 位

五、总结

我们的车牌识别系统基于 Matlab 计算平台开发，提供具有图形用户界面的可执行程序实现使用交互。

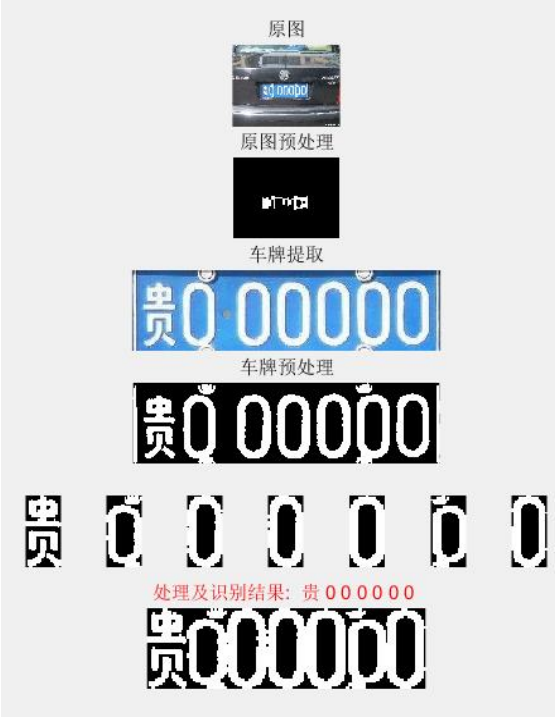
算法部分分五大模块：原始图像预处理、车牌区域图像提取、车牌区域图像预处理，文字区域裁切，另含辅助功能模块若干，均采用函数封装。依赖项包含字符识别模板。

系统输入为简单的静态图像，输出为识别结果及部分过程信息。

静态图像车牌识别的核心难点在于如何从图片中准确地寻找到有效区域并恰到好处地实现切割。为此，我们在图像预处理的模块上做了大量尝试，为尽可能地提高程序准确率和普适性反复改进调用算法和相关参数，例如，采用了大津法(OTSU)实现高鲁棒性的阈值分割，运用了形态学的腐蚀、闭合并精心调试了结构元素尺寸参数……

字符识别方面，考虑到车牌采用法定统一标准，具有严格规定的格式和字体，因此，字符识别部分采用模板匹配是一个可靠合理且简单易行的方案。

目前，本车牌识别系统能够抽取并识别一些画面较为简单的车牌图像，有些图像受限与画面倾斜等因素，能够提取切割，而文字识别率可能不高。受限于图像预处理算法无法动态适应复杂的画面场景，难以从静态图像中完整滤除杂讯，准确提取出车牌区域及切割出标准的字符像素域，与商用级别的视频流识别仍有差距。



附件：

附 1：GUI 程序交互展示

a ） 选择待识别图片

b ） 点击 [开始识别] 即可，结果如图所示。

附 2：图例（主函数例程，开发调试用，运行截图）

附 3：代码页

```
function [] = main(~)

% 0、主函数例程（开发调试用）
% function [] = main(~)
% 负责人：沈家赟
%
% 说明：
% 输入     原图
% 输出     识别过程信息，识别结果
%
% 功能
%  此主函数Matlab脚本实现系统整体调用，作为开发调试使用，
%  包含整体系统架构、顶层注释并在结尾提供逐模块的过程显示

clear
close all
clc

% 1) 读取原图
[filename, filepath] = uigetfile('*.jpg','选择图片');
url_Img = strcat(filepath, filename);
I_raw = imread(url_Img);
%%
%%%%%%%%%%%%%%% 原始图像预处理 %%%%%%%%%%%%%%%%

I_Proced = preProcRawImg(I_raw);

%%
%%%%%%%%%%%%%%% 定位裁剪车牌图像 %%%%%%%%%%%%%%%%

I_plateRaw = getPlateImg(I_Proced, I_raw);

%%
%%%%%%%%%%%%%%% 车牌图像预处理 %%%%%%%%%%%%%%%%

I_plateProced = preProcPlateImg(I_plateRaw);

%%
%%%%%%%%%%%%%%% 车牌文字裁切 %%%%%%%%%%%%%%%%
% 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
```

```
[word1, word2, word3, word4, word5, word6, word7] =
partitionWords( I_plateProced );

%%
%%%%%%%%%%%%%% 文字识别 %%%%%%%%%%%%%%%%

Code = recognizeWords( word1, word2, word3, word4, word5, word6, word7 );

%%
figure(1);
subplot(6,7,(1:7)), imshow(I_raw),title('原图');
subplot(6,7,(8:14)), imshow(I_Proced),title('原图预处理');
subplot(6,7,(15:21)), imshow(I_plateRaw),title('车牌提取');
subplot(6,7,(22:28)), imshow(I_plateProced),title('车牌预处理');
subplot(6,7,29), imshow(word1);
subplot(6,7,30), imshow(word2);
subplot(6,7,31), imshow(word3);
subplot(6,7,32), imshow(word4);
subplot(6,7,33), imshow(word5);
subplot(6,7,34), imshow(word6);
subplot(6,7,35), imshow(word7);
subplot(6,7,(36:42)), imshow([word1, word2, word3, word4, word5, word6,
word7]),title (['处理及识别结果:  ', Code],'Color','red');
```

```
function I_out = preProcRawImg( Img_raw )

% 1、原始图像预处理 【技术难点】
% I_Proced = preProcRawImg(I_raw)
% 负责人：吴晓灿
%
% 说明：
% 函数    preProcRawImg
% 输入    原图
% 输出    能够区分车牌和背景的二值图
%
% 原理
% 首先将原图像转化为灰度图，然后选定合适的阈值，将灰度图转化为二值图。
```

```
% 用roberts算子进行边缘检测，对边缘图像进行形态学腐蚀，
% 之后进行合理地形态学闭合，
% 最后将图像中的小对象移除，得到目标车牌所处的位置二值图。


%%%%%%%%%%%%%%% 原始图像预处理 %%%%%%%%%%%%%%%%
%1)原图转换灰度图
I1_rgb2gray = rgb2gray(Img_raw);

%2)灰度图转换二值图
%===========改进=============
T=graythresh(I1_rgb2gray);
I2_binary=im2bw(I1_rgb2gray,T);

%3)边缘检测：sobel
I3_edge = edge(I2_binary,'sobel',0.15);

%4) 腐蚀图像: 形态学腐蚀
se = [1;1];
I4_erosion = imerode(I3_edge,se);

%5)平滑图像轮廓：形态学闭合
se = strel('rectangle',[24,24]);    % 形态学结构元素
%=================参数需调节！！！！！！！！！ ============================
%车牌处矩形面积较小时，[]内数值调大
I5_smooth = imclose(I4_erosion,se);


% 附：经验设计，上下左右边界绝大多数情况下均为杂讯（沈家赟）
I_cutmargin = I5_smooth;
[m,n,~] = size(I_cutmargin);
vertical_margin = m/10      % 清除垂直方向边界，即上下边界
horizontal_margin = n/10    % 清除水平方向边界，即左右边界（注意：n/10较为
激进）
for i = 1:m
    for j = 1:n
        if (i<vertical_margin || i>m-vertical_margin || j<horizontal_margin
|| j>n-horizontal_margin)
            I_cutmargin(i,j) = 0;
        end
    end
end
I5_smooth = I_cutmargin;
```

```
%6)从对象中移除小对象
imLabel = bwlabel(I5_smooth);                    %对各连通域进行标记
stats = regionprops(imLabel,'Area');            %求各连通域的大小
area = cat(1,stats.Area);                       %建立索引
index = find(area == max(area));                %求最大连通域的索引
I6_filter = ismember(imLabel,index);             %获取最大连通域图像

I_out = I6_filter;
end
```

```
function I_out = getPlateImg( Img_preproc, Img_raw )

% 2、定位裁剪车牌图像
% I_plateRaw = getPlateImg(I_Proced, I_raw)
% 负责人：戴星辰
%
% 说明：
%   输入:预处理后的二值化图像与原图
%   输出：裁剪后的拆牌图像
%   图像定位与裁剪，即给定原图与对应的已与处理的二值化图像来裁剪出正确的车牌位置。算法过程：
%   1.将预处理后的图像表明连通区域
%   2.合并相似的连通区域（某些时候车牌会被处理成不同的连通域）
%   3.查找长宽比最佳的连通域
%   4.对已查找到的连通域进一步细分四个边界（通过判断每一行列的非零像素点数）
%   5.根据边界从原图割出车牌区域。

%%%%%%%%%%%%%%% 定位裁剪车牌图像 %%%%%%%%%%%%%%%%
[row, col]=size(Img_preproc);
I7_double = double(Img_preproc);
vector=getBestArea(I7_double)
up=vector(1)
down=vector(2)
left=vector(3)
right=vector(4)
```

```
 %%%%%% ROWs: 逐行扫描 %%%%%%%%% 车牌区域逐行像素个数统计
 Plate_PixelsPerRow = zeros(row,1);
 for index_row = up:down
    for index_col = left:right
         if(I7_double(index_row,index_col,1)==1)
            Plate_PixelsPerRow(index_row,1) =
Plate_PixelsPerRow(index_row,1)+1;%蓝色像素点统计
        end
     end
 end
 [~, indexOfMaxRowPixels]=max(Plate_PixelsPerRow);%车牌区域-最密集行位置确
定

 % 自车牌像素最密集行开始，向上寻找车牌上边界
 plate_row_top = indexOfMaxRowPixels;
 while ((Plate_PixelsPerRow(plate_row_top,1)>=5)&&(plate_row_top>1))
        plate_row_top = plate_row_top-1;
 end

 % 自车牌像素最密集行开始，向上寻找车牌下边界
 plate_row_bottom = indexOfMaxRowPixels;
 while
((Plate_PixelsPerRow(plate_row_bottom,1)>=5)&&(plate_row_bottom<row))
        plate_row_bottom = plate_row_bottom+1;
 end

 I_rowcut = Img_raw(plate_row_top:plate_row_bottom,:,:);

 %%%%%% COLs: 逐列扫描 %%%%%%%%% 车牌区域逐列像素个数统计
 Plate_PixelsPerCol = zeros(1,col); % 进一步确定列方向的车牌区域
 for index_col = left:right
     for index_row = plate_row_top:plate_row_bottom
        if(I7_double(index_row,index_col,1)==1)
            Plate_PixelsPerCol(1,index_col)=
Plate_PixelsPerCol(1,index_col)+1;
        end
     end
 end

% 自最左列开始，向右寻找车牌左边界
plate_col_left = 1;
while ((Plate_PixelsPerCol(1,plate_col_left)<3)&&(plate_col_left<col))
   plate_col_left = plate_col_left+1;
```

```
end
% 自车牌像素最密集列开始，向左寻找车牌左边界
plate_col_right = col;
while
((Plate_PixelsPerCol(1,plate_col_right)<3)&&(plate_col_right>plate_col_left
))
    plate_col_right = plate_col_right-1;
end

plate_col_left=plate_col_left-1;%对车牌区域的校正
plate_col_right=plate_col_right+1;

I_colcut = Img_raw(:,plate_col_left:plate_col_right,:);

up_shrink = int32((plate_row_bottom-plate_row_top)/20)
bottom_shrink = int32((plate_row_bottom-plate_row_top)/7)

I8_plate=Img_raw(plate_row_top+up_shrink:plate_row_bottom-bottom_shrink,
plate_col_left:plate_col_right,:);

I_out = I8_plate;
end
```

```
function [ vector ] = getBestArea( Img_preproc )
%GETMATRIX 此处显示有关此函数的摘要
%   此处显示详细说明
T_double = double(Img_preproc);
[row, col]=size(Img_preproc);
[L,num]=bwlabel(T_double, 4);%获得连通区域与个数%
matrixs=zeros(num,4)%构造对应大小矩阵%

%定位每个联通区域的上下左右界%
for i=1:num
    temp=-1;
    for index_row=1:row
```

```
        for index_col=1:col
            if (L(index_row,index_col)==i)
                if(temp==-1)
                    matrixs(i,1)=index_row;%第一次检测到上界%
                    temp=0;
                else
                    temp = temp+1;
                end
                break
            end
        end
    end
    matrixs(i,2)=matrixs(i,1)+temp;%上界+偏移=下界%
    temp = -1;
    for index_col=1:col
        for index_row=1:row
            if (L(index_row,index_col)==i)
                if(temp==-1)
                    matrixs(i,3)=index_col;
                    temp=0;
                else
                    temp = temp+1;
                end
                break
            end
        end
    end
    matrixs(i,4)=matrixs(i,3)+temp;
end

%判断上下界相似度  是否合并%
if(num==1)
    vector=matrixs(1,:)
    return
end
for i=1:num
    for j=i+1:row
        if (j>num)
            break
        end
        %%如果上界与下界几乎相同 可视为一个整体
        if(abs(matrixs(i,1)-matrixs(j,1))<8&&abs(matrixs(i,2)-
matrixs(j,2))<8)
            matrixs(i,3)=min([matrixs(i,3),matrixs(j,3)]);
```

```
            matrixs(i,4)=max([matrixs(i,4),matrixs(j,4)]);
            matrixs(j,:)=[-1,-1,-1,-1];
            num=num-1
        end
    end
end

%再次检索长与宽之比是否符合要求%
if(num==1)
    vector=matrixs(1,:)
    return
end
best=1
best_pro=9%当前最佳比例
for i=1:num
    if(abs(3.142-(matrixs(i,4)-matrixs(i,3))/(matrixs(i,2)-
matrixs(i,1)))<best_pro)
        %更新最佳区域%
        best=i
        best_pro=abs(3.142-(matrixs(i,4)-matrixs(i,3))/(matrixs(i,2)-
matrixs(i,1)))
    end
end
vector=matrixs(best,:);
```

```
function I_out = preProcPlateImg( Img_plate )

% 3、车牌图像预处理
% I_plateProced = preProcPlateImg(I_plateRaw)
% 负责人：陆佳程
%
% 说明：
% 输入为较为清楚的车牌周围的图像
% 输出为细化后的车牌二值图像
%
% 原理：
% 首先对输入图像进行二值化处理，然后利用robert算子边缘检测，
% 检测出边缘后进行腐蚀然后平滑。处理后车牌部分是一个比较大的对象，
% 把小对象移除后就能确定出车牌的坐标。利用坐标从原图中将车牌截出，
```

```
% 然后就利用二值化、均值滤波、膨胀，就能得到预处理后的车牌图像。

%%%%%%%%%%%%%% 车牌图像预处理 %%%%%%%%%%%%%%%%
% [filename,filepath]=uigetfile('cuttedplate.jpg','选择裁剪出的车牌图像');
% url_plate=strcat(filepath,filename);
% a=imread(url_plate);
a = Img_plate;
b=rgb2gray(a);

g_max=double(max(max(b)));
g_min=double(min(min(b)));
T=round(g_max-(g_max-g_min)/3); % T 为二值化的阈值
[m,n]=size(b);
d=(double(b)>=T);  % d:二值图像

% 滤波
h=fspecial('average',3);
d=im2bw(round(filter2(h,d)));

% 某些图像进行操作
% 膨胀或腐蚀
% se=strel('square',3);  % 使用一个3X3的正方形结果元素对象对创建的图像进行
膨胀
% 'line'/'diamond'/'ball'...
se=eye(2); % eye(n) returns the n-by-n identity matrix 单位矩阵
[m,n]=size(d);
if bwarea(d)/m/n>=0.365
    d=imerode(d,se);
elseif bwarea(d)/m/n<=0.235
    d=imdilate(d,se);
end

% 对预处理后的车牌文字区域再做收缩切割，即更新车牌区域
I_out = minimizeDistrict(d);

end
```

```
function [word1, word2, word3, word4, word5, word6, word7] =
partitionWords( I_plateProced )

% 4、车牌文字裁切
%   [word1,word2,word3,word4,word5,word6,word7] =
partitionWords(I_plateProced)
% 负责人：高冰
%
% 说明：
% 输入输出：输入一个二值图像，输出对应的字符模板。
%
% 过程：
%   首先循环找出字符区域，判断字符区域大小，对于包含多个字符的区域，找出像
素值最少的列，清空像素值，分开字符区域。
%   开始依次分割字符区域，并调用getword函数获取相对应的字符模板。
%   其中对于第一个字符区域，判断是否是字符，如果小于经验宽度，就不是字符，
清空该区域像素值，进行切割。直至找到符合宽度的字符区域，并且该区域中间的像
素值总和大于经验最小值，就认为是第一个字符，分割成功。
%   输出对应的字符模板。

%%%%%%%%%%%%%% 车牌文字裁切 %%%%%%%%%%%%%%%%
% 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 = I_plateProced;

[m,n]=size(d);

k1=1;%某个字符区域的最左边
k2=1;%某个字符区域的最右边

s=sum(d);%行向量，每列像素值的和

i=1;%循环计数变量
while i~=n

    %找出某个字符区域的最左边
    while s(i)==0
        i=i+1;
    end
    k1=i;

    %找出某个字符区域的最右边
    while s(i)~=0 && i<=n-1
```

```
        i=i+1;
    end
    k2=i-1;

    if k2-k1>=round(n/6.5)%判断该字符区域宽度，确定是不是包含了多个字符
        [val,num]=min(sum(d(:,[k1+5:k2-5])));%找出字符间隔，即像素值和最少的那一列
        d(:,k1+5+num)=0;  % 清空 字符间隔区域的像素值
    end
end

d=minimizeDistrict(d); %缩小、细化区域
%%切割出 7 个字符
%一些经验数值
y1=10;
y2=0.25;
%信号变量
flag=0;

%分割出第一个字符，因为有左侧干扰，首先消除左侧干扰
word1=[];
while flag==0
    [m,n]=size(d);
    %找出 可能 包含某字符区域最右边的列号，也可认为就是 此区域的宽度
    wide=0;
    while sum(d(:,wide+1))~=0
        wide=wide+1;
    end

    if wide<y1   % 如果 最右边列号 小于 给定界限，即 此区域宽度 小于 经验宽度，认为是 左侧干扰,不是 字符
        d(:,[1:wide])=0;%清空干扰的像素值
        d=minimizeDistrict(d);%切割
    else
        temp=minimizeDistrict(imcrop(d,[1 1 wide m]));%裁剪字符区域
        [m,n]=size(temp);
        all=sum(sum(temp));%总的像素值
        two_thirds=sum(sum(temp([round(m/3):2*round(m/3)],:)));%字符区域中间部分的像素值之和

        if two_thirds/all>y2%如果 字符区域中间部分的像素值之和 所占 总像素值 比重 大于经验最小值，则认为 此部分 包含字符
            flag=1; %结束信号
            word1=temp;    %保存区域
```

```
        end
        d(:,[1:wide])=0;d=minimizeDistrict(d);
    end
end


% 分割出第二个字符
[word2,d]=getword(d);
% 分割出第三个字符
[word3,d]=getword(d);
% 分割出第四个字符
[word4,d]=getword(d);
% 分割出第五个字符
[word5,d]=getword(d);
% 分割出第六个字符
[word6,d]=getword(d);
% 分割出第七个字符
[word7,d]=getword(d);

%[m,n]=size(word1);
% 统一化大小为 40*20后显示
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]);

end
```

```
function Code = recognizeWords( word1, word2, word3, word4, word5, word6,
word7 )

% 5、文字识别 【技术难点】
% Code = recognizeWords( word1, word2, word3, word4, word5, word6, word7 )
```

```
% 负责人：张珍茹， 魏然
%
% 说明：
% 字符识别是车牌识别中很重要的一部分，在模式识别中也扮演的很重要的角色。这里使用的是模板匹配算法：
% 1、首先本文识别的前提是模板字符是二值图像，背景黑色，字符为白色；同样待识别的字符同样如此。
% 2、将预处理后的待识别字符图像imageU与字符模板库中的字符图像imageT进行"与"运算得到共同部分图像imageV；
% 3、将得到的共同部分图像与待识别字符进行逻辑”异或“运算，得到待识别字符图像多余部分imageX；
% 4、将得到的共同部分图像与模板字符进行逻辑"异或"运算，得到模板图像多余部分imageW。
% 5、计算每个模板字符图像imageT中白像素个数T,待识别字符图像imageU的白像素个数U，
% 6、imageU与imageT共同的部分imageV的白像素个数V，imageW的白像素个数W；imageX中白像素个数X；
% 7、构造判别函数表达式为：
%     tempSum=sqrt(((T-TUV)*(T-TUV)+(U-TUV)*(U-TUV)+(V-TUV)*(V-TUV))/2);
%     Y(j)=V/(W/T*X/U*tempSum);
%     其中TUV=（T+U+V）/3;
%     这样，相似系数最大max（Y）对应的模板M为待识别字符；

templatePath='templateCharacter\';
fileFormat='.bmp';
templateImage=zeros(40,20,67);
%Timage=zeros(67,800);
for i=1:67 %读取模板
    stri=num2str(i-1);
    imagePath=[templatePath,stri,fileFormat];
    tempImage=imread(imagePath);
    templateImage(:,:,i)=tempImage;
    clear imagePath stri tempImage;
end

%characterImage(40,20,7)= {word1,word2,word3,word4,word5,word6,word7};
characterImage = zeros(40,20,7);
 characterImage(:,:,1)= word1;
 characterImage(:,:,2)= word2;
 characterImage(:,:,3)= word3;
 characterImage(:,:,4)= word4;
 characterImage(:,:,5)= word5;
 characterImage(:,:,6)= word6;
 characterImage(:,:,7)= word7;
```

```
%Uimage=zeros(7,800);

liccode=char(['0':'9' 'A':'Z' '京冀津晋蒙辽吉黑沪苏浙皖闽赣鲁豫鄂湘粤桂琼渝川贵云藏陕甘青宁新']);
Y=zeros(1,67);
I=1;
for i=1:7
    U=length(find( characterImage(:,:,i))~=0);%计算待识别字符中白像素的个数
    for j=1:67
        T=length(find( templateImage(:,:,j))~=0); %计算字符模板中白像素的个数
        tempV=characterImage(:,:,i)& templateImage(:,:,j); %求字符模板与待测图像的并集
        V=length(find(tempV)~=0);
        tempW=xor(tempV,templateImage(:,:,j)); %求字符模板的多余部分
        W=length(find(tempW)~=0);
        tempX=xor(tempV,characterImage(:,:,i)); %求待测图像的多余部分
        X=length(find(tempX)~=0);
        %构造判别函数
        TUV=(T+U+V)/3;
        tempSum=sqrt(((T-TUV)*(T-TUV)+(U-TUV)*(U-TUV)+(V-TUV)*(V-TUV))/2);
        Y(j)=V/(W/T*X/U*tempSum);
    end
    [MAX,indexMax]=max(Y); %相似系数最大max（Y）对应的模板M为待识别字符
    Code(I*2-1)=liccode(indexMax);
    Code(I*2)=' ';
    I=I+1;

    clear imagePath indexMax;
end
```

```
function [word,result]=getword(d)

% 6、提取车牌文字区域
% [word, result] = getword(d)
% 负责人：封琳
```

```
%
% 说明：
% 在汽车牌照自动识别过程中，字符分割有承前启后的作用。它在前期牌照定位的基础上进行字符的分割，然后再利用分割的结果进行字符识别。函数function [word,result]=getword(d)用来对预处理后的车牌文字区域做提取，从车牌中分割出第n个字符。
% （1）    开始的d为已经预处理过的车牌图像
% （2）    imcrop是一个函数，在MATLAB中，该函数用于返回图像的一个裁剪区域。
% [A,rect] = imcrop(...)格式指定了要裁剪的区域。imcrop(d,[1 1 wide m])为从坐标（1,1）处开始，裁剪出大小为（wide,m）的区域
% （3）    size() 函数：获取矩阵的行数和列数
% [r,c]=size(A)当有两个输出参数时，size函数将矩阵的行数返回到第一个输出变量r，将矩阵的列数返回到第二个输出变量c。
% （4）    sum(A; 列求和，以矩阵A的每一列为对象，对一列内的数字求和。
% sum(A,2); 行求和，以矩阵A的每一行为对象，对一行内的数字求和。
% sum(A(:)); 对矩阵进行全矩阵数值求和
% （5）    如果切除的字符块 大于y1,且宽高比为1:2 ，就认为是有效字符 ，进行切割。


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
            wide=wide+1;
        end
        temp=minimizeDistrict(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=minimizeDistrict(d);  % 切割出最小范围
            else word=[];flag=1;
            end
        else
            word=minimizeDistrict(imcrop(d,[1 1 wide m]));
            d(:,[1:wide])=0;
            if sum(sum(d))~=0;
                d=minimizeDistrict(d);flag=1;
            else d=[];
            end
        end
    end
```

```
%end
          result=d;
```

```
function I_out = minimizeDistrict( I_in )

% 7、收缩、细化有效区域
% I_out = minimizeDistrict(I_in)
% 负责人：丁瑞宇
%
% 说明：
% 对预处理后的车牌文字区域再做收缩切割，即更新车牌区域。

% 对预处理后的车牌文字区域再做收缩切割，即更新车牌区域
[m,n]=size(I_in);
top=1;
bottom=m;
left=1;
right=n;   % init
while sum(I_in(top,:))==0 && top<=m
    top=top+1;
end
while sum(I_in(bottom,:))==0 && bottom>=1
    bottom=bottom-1;
end
while sum(I_in(:,left))==0 && left<=n
    left=left+1;
end
while sum(I_in(:,right))==0 && right>=1
    right=right-1;
end
I_out = I_in(top:bottom, left:right, :);

end
```

```
function varargout = Licenseplaterecognition(varargin)
% LICENSEPLATERECOGNITION M-file for Licenseplaterecognition.fig
%      LICENSEPLATERECOGNITION, by itself, creates a new
LICENSEPLATERECOGNITION or raises the existing
%      singleton*.
%
%      H = LICENSEPLATERECOGNITION returns the handle to a new
LICENSEPLATERECOGNITION or the handle to
%      the existing singleton*.
%
%      LICENSEPLATERECOGNITION('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in LICENSEPLATERECOGNITION.M with the given
input arguments.
%
%      LICENSEPLATERECOGNITION('Property','Value',...) creates a new
LICENSEPLATERECOGNITION or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before Licenseplaterecognition_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to Licenseplaterecognition_OpeningFcn
via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
Licenseplaterecognition

% Last Modified by GUIDE v2.5 30-Dec-2016 20:54:01

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn',
```

```
@Licenseplaterecognition_OpeningFcn, ...
                   'gui_OutputFcn',
@Licenseplaterecognition_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
% End initialization code - DO NOT EDIT


% --- Executes just before Licenseplaterecognition is made visible.
function Licenseplaterecognition_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Licenseplaterecognition (see
VARARGIN)

% Choose default command line output for Licenseplaterecognition
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Licenseplaterecognition wait for user response (see
UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Licenseplaterecognition_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in btn_load.
function btn_load_Callback(hObject, eventdata, handles)
% hObject    handle to btn_load (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global in;                  %定义一个全局变量in
[filename, filepath] = uigetfile('*.jpg','选择图片');
url_Img = strcat(filepath, filename);
in=imread(url_Img);             %读取图像
axes(handles.axes1);        %使用第一个axes
cla(handles.axes1,'reset');
cla(handles.axes2,'reset');
cla(handles.axes3,'reset');
cla(handles.axes4,'reset');
cla(handles.axes5,'reset');
cla(handles.axes6,'reset');
cla(handles.axes7,'reset');
cla(handles.axes8,'reset');
cla(handles.axes9,'reset');
cla(handles.axes10,'reset');
cla(handles.axes11,'reset');
cla(handles.text4,'reset');
cla(handles.text5,'reset');
cla(handles.text6,'reset');
cla(handles.text7,'reset');
cla(handles.text8,'reset');
cla(handles.text9,'reset');
cla(handles.text10,'reset');
imshow(in);                 %显示图像

% --- Executes on button press in btn_recognize.
function btn_recognize_Callback(hObject, eventdata, handles)
% hObject    handle to btn_recognize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global in;                  %定义一个全局变量in
I_raw = in;
%%
%%%%%%%%%%%%%% 原始图像预处理 %%%%%%%%%%%%%%%%
```

```
I_Proced = preProcRawImg(I_raw);

%%
%%%%%%%%%%%%%%% 定位裁剪车牌图像 %%%%%%%%%%%%%%%%

I_plateRaw = getPlateImg(I_Proced, I_raw);

%%
%%%%%%%%%%%%%%% 车牌图像预处理 %%%%%%%%%%%%%%%%

I_plateProced = preProcPlateImg(I_plateRaw);

%%
%%%%%%%%%%%%%%% 车牌文字裁切 %%%%%%%%%%%%%%%%
% 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

[word1, word2, word3, word4, word5, word6, word7] =
partitionWords( I_plateProced );

%%
%%%%%%%%%%%%%%% 文字识别 %%%%%%%%%%%%%%%%

Code = recognizeWords( word1, word2, word3, word4, word5, word6, word7 );

%%
axes(handles.axes2);        %使用第二个axes
imshow(I_Proced);
axes(handles.axes3);        %使用第三个axes
imshow(I_plateRaw);
axes(handles.axes4);        %使用第四个axes
imshow(I_plateProced);

axes(handles.axes5);        %使用第五个axes
imshow(word1);
axes(handles.axes6);        %使用第六个axes
imshow(word2);
axes(handles.axes7);        %使用第七个axes
imshow(word3);
axes(handles.axes8);        %使用第八个axes
imshow(word4);
axes(handles.axes9);        %使用第九个axes
imshow(word5);
axes(handles.axes10);        %使用第十个axes
```

```
imshow(word6);
axes(handles.axes11);          %使用第十一个axes
imshow(word7);

set(handles.text4,'string',Code(1));
set(handles.text5,'string',Code(3));
set(handles.text6,'string',Code(5));
set(handles.text7,'string',Code(7));
set(handles.text8,'string',Code(9));
set(handles.text9,'string',Code(11));
set(handles.text10,'string',Code(13));
```