**网络空间安全学院**

数字图像处理课程设计报告

车牌自动识别算法的设计与实现

|  |  |  |  |
| --- | --- | --- | --- |
| 姓名1： | 杨建源 | 姓名2： | 陈钿 |
| 学号1： | 201741404250 | 学号2： |  |
|  |  |  |  |
| 日期： |  |  |  |

目录

[一、设计任务说明 1](#_Toc270294099)

[1.1 课程设计任务 1](#_Toc905008488)

[1.2 开发该系统软件环境及使用的技术说明 1](#_Toc1998469762)

[二、系统设计 1](#_Toc1637050854)

[2.1 系统基本工作流程 1](#_Toc353217814)

[2.2 图像分割算法设计 1](#_Toc886999590)

[2.3 图像增强算法设计 1](#_Toc2118115303)

[2.4车牌识别算法说明 1](#_Toc327481202)

[2.5 用户接口设计 1](#_Toc2123458400)

[三、系统实施及结果 1](#_Toc2082082954)

[3.1系统操作说明 1](#_Toc322180013)

[3.2 算法性能 1](#_Toc1073204404)

[3.3 github提交日志 1](#_Toc631266875)

[四、课程设计总结 2](#_Toc1133151945)

**一、设计任务说明**

1.1 课程设计任务

利用OpenCV库和其他开源工具，设计并实现车牌自动识别算法，基本功能要求：

1. 对给定的包含有汽车车牌的照片进行处理，利用图像分割算法将目标从背景中分离出来。
2. 对目标图像进行合适的处理，然后利用Tesseract库实现车牌号码的识别，将结果输出。
3. 要求提供比较友好的用户接口，可以对新的图片导入到系统中进行处理，并将结果返回给用户。
4. 利用C／C++代码实现，推荐Linux系统，也可以在Windows／Mac系统下完成。
5. 要求处理过程的自动化，即输入图像，自动输出车牌信息，无需人去干预。

1.2 开发该系统软件环境及使用的技术说明

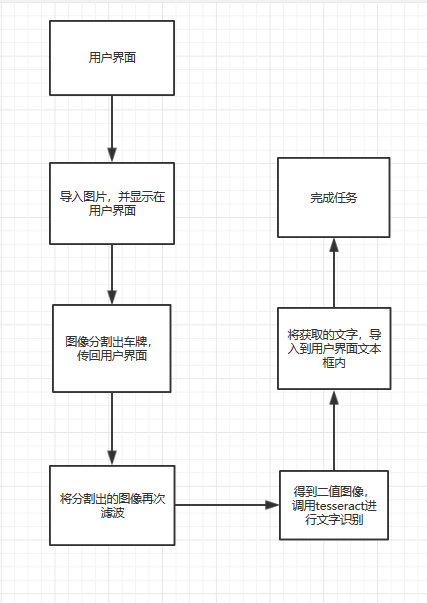
开发环境选择了windows+opencv+tesseract+qt

1.3 分工情况

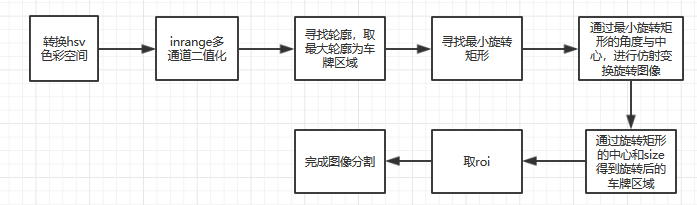
杨建源同学负责图像分割和tesseract的调用，陈钿同学负责用户界面的设计。

1. **系统设计**

2.1 系统基本工作流程



2.2 图像分割算法设计



cvtColor(src, binary\_output, CV\_BGR2HSV); //转换色彩空间

inRange(binary\_output, Scalar(80, 160, 103), Scalar(180, 255, 255), binary\_output);

//二值化

findContours(binary\_output, contours, hierarcy, CV\_RETR\_EXTERNAL, CV\_CHAIN\_APPROX\_NONE); //寻找轮廓

for (int i = 0; i<contours.size(); i++) //寻找最大轮廓

if(contourArea(contours[i])>=contourArea(contours[max]))

max=i;

RotatedRect rect = minAreaRect(contours[max]); //画最小矩形

Size dst\_sz(src.cols, src.rows);

cv::Mat rot\_mat = cv::getRotationMatrix2D(rect.center, rect.angle, 1.0);

cv::warpAffine(src2, src2, rot\_mat, dst\_sz); //仿射变换，进行旋转

src3=src2(Rect((rect.center.x-(rect.size.width/2)),(rect.center.y-(rect.size.height/2)),rect.size.width,rect.size.height)); //取roi，得到车牌

2.3 图像增强算法设计

cvtColor(src3, src3, CV\_BGR2HSV);

inRange(src3, Scalar(0, 0, 130), Scalar(180, 130, 255), src3); //二值化图像

Mat element = getStructuringElement(MORPH\_RECT, Size(3, 3));

morphologyEx(src3, src3, MORPH\_OPEN, element); //进行开操作，消除噪点

subtract(Scalar(255,255,255),src3,src3); //取反，使得图像白底黑字便于文字识别

## 2.4车牌识别算法说明

接口函数是通过工程调用终端命令(tesseract xxx.JPG stdout -l chi\_sim)，进行识别，然后捕获终端输出，将其显示在用户界面上。(本来是想调用tesseract的api的，但是编译的时候需要先编译tiff与leptonica。后者需要翻墙下源码，就选择直接通过命令行进行调用。)

int \_System(const char \* cmd, char \*pRetMsg, int msg\_len)

{

FILE \* fp;

char \* p = NULL;

int res = -1;

if (cmd == NULL || pRetMsg == NULL || msg\_len < 0)

{

printf("Param Error!\n");

return -1;

}

if ((fp = \_popen(cmd, "r")) == NULL)

{

printf("Popen Error!\n");

return -2;

}

else

{

memset(pRetMsg, 0, msg\_len);

while (fgets(pRetMsg, msg\_len, fp) != NULL&&pRetMsg[0]==' ');

if ((res = \_pclose(fp)) == -1)

{

printf("close popenerror!\n");

return -3;

}

pRetMsg[strlen(pRetMsg) - 1] = '\0';

return 0;

}

}

2.5 用户接口设计

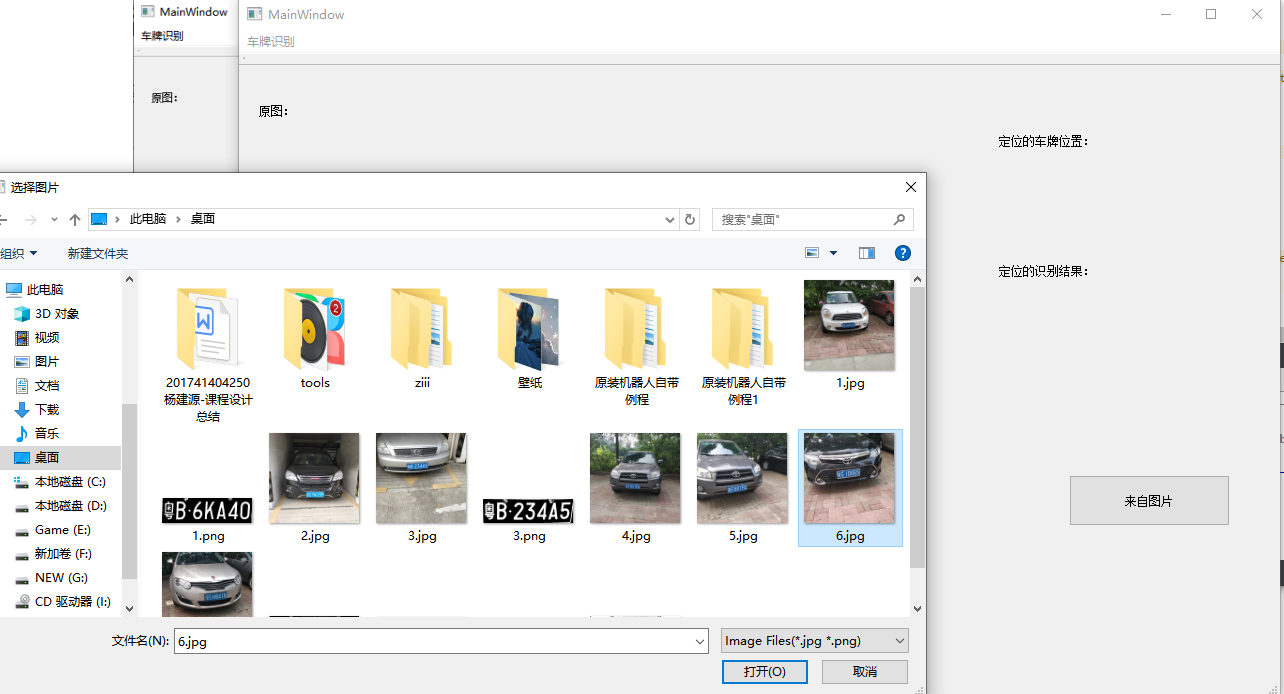
接口设计思路，相关代码

**三、系统实施及结果**

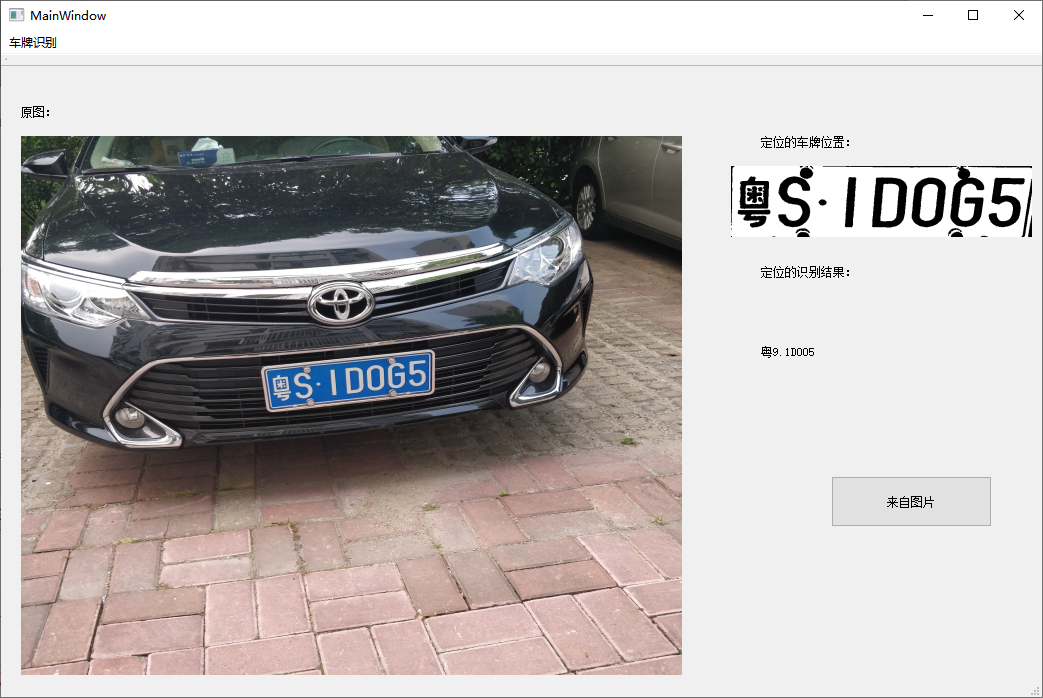
## 3.1系统操作说明

## 

初始界面



点击按钮选择图片



得到结果

## 3.2 算法性能

## 

## 图像处理用了0.393秒(由于太多整张图片为尺度的处理,且因为图像分辨率很高，为了处理的准确性没有降低分辨率，所以得到这个时间)

准确性：本次调用的tesseract库为原生的中文库(经过多种方案的尝试后的选择，具体详见课程设计总结处)，存在误差与干扰挺大的。

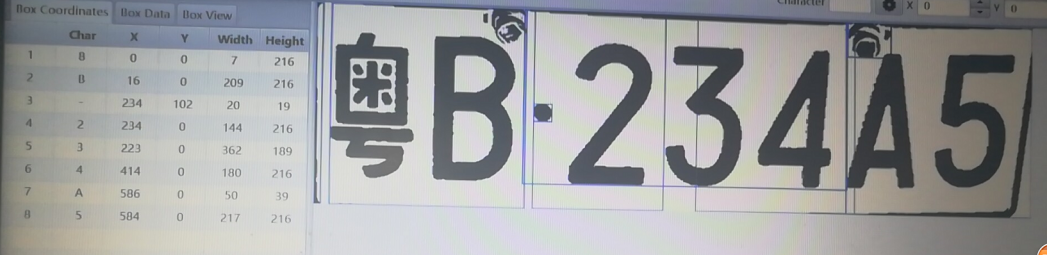
## 3.3 github提交日志

截图显示各个成员在github提交的日志

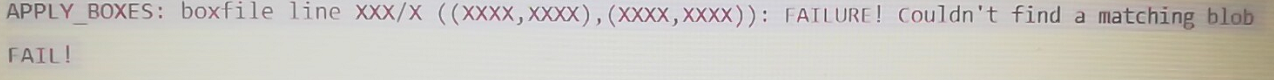
**四、课程设计总结**

①关于tesseract的调用问题，由于是在windows环境下进行开发，使用cmake编译tesseract的时候，需要先编译lib和leptonica，这里由于找不到后者的资源，就选择了使用安装包，然后使用命令行进行tesseract功能的实现以及调用，但两者的结果是没有差异的

②关于ocr准确率的问题，官方原生的中文库存在的问题挺大的，所以按理来说应该自己训练，刚好官方有提供jTessBoxEditorFX方便进行训练，然后问题就出现了，一是得到的.tif格式的图片(就是经过自动分割处理的图片),原本自己的分割出现了很严重的偏差，疑惑的地方是这么凌乱的切割得到的结果是正确的(除了中文)。这里没有多管，



直接进行训练的操作，直接手动切割每一个字符,最后训练的时候出现了匹配块不匹配的问题(后面发现只要移动了这看似凌乱的切割，就会出现错误，但是不移动训练就没什么意义了)，途中也测试过更换背景色的操作，结果还是一样。



最后选择寻找其他的中文ocr，如百度云，腾讯云，但是，无一例外，都是要收费的，摸摸我的口袋选择了放弃...

总的来说，这次的课程设计学习到了关于图像分割与图像增强的相关知识，并部署到了一个应用程序上。算是一个完整性还可以的课程设计吧。但关于图像的相关操作，仍让需要进一步的学习与补充，如怎么克服多种不同环境的影响，这次课程设计没有做太多这方面的考虑(仅仅基于原有的图片进行操作)，包括到如果是视频环境的如何操作，这些都是需要继续发展学习的地方。