**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： | 梁梓豪 | 姓名2： | 陈继伟 |
| 学号1： | 201741404139 | 学号2： | 201741404138 |
|  |  |  |  |
| 日期： | 2019.6.18 |  |  |

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 2](#_Toc1637050854)

[2.1 系统基本工作流程 2](#_Toc353217814)

[2.2 图像分割算法设计 2](#_Toc886999590)

[2.3 图像增强算法设计 3](#_Toc2118115303)

[2.4车牌识别算法说明 3](#_Toc327481202)

[2.5 用户接口设计 4](#_Toc2123458400)

[三、系统实施及结果 5](#_Toc2082082954)

[3.1系统操作说明 5](#_Toc322180013)

[四、课程设计总结 5](#_Toc1133151945)

1. **设计任务说明**

**1.1 课程设计任务**

1. 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：
2. 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。
3. 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。
4. 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。
5. 利用C／C++代码实现，推荐Linux系统，也可以在Windows／Mac系统下完成。
6. 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**

环境：win10、visual studio2017

技术：Opencv3.4、qt5.1、Tesseract4.1.0

**1.3 分工情况**

梁梓豪：识别车牌算法设计及opencv、Tesseract的配置。

陈继伟：qt设计及连接槽、读取图片

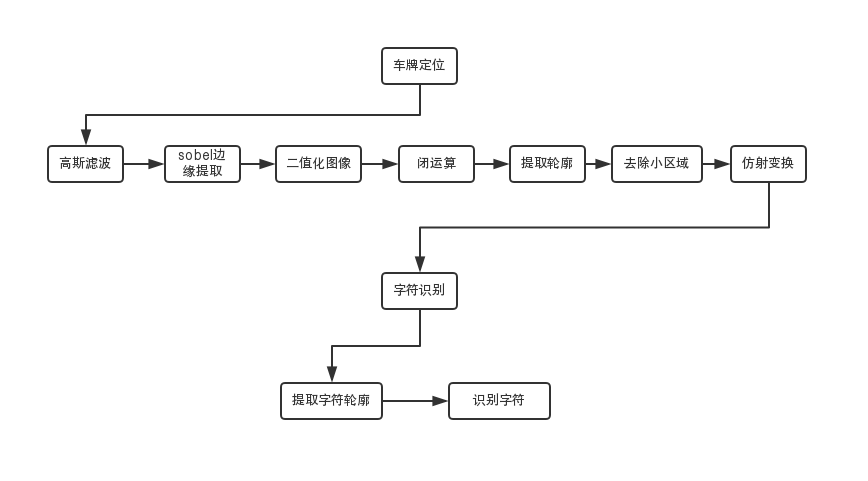
**二、系统设计**

2.1 系统基本工作流程

第一，先对原始图像进行增强，方便查找车牌位置。

第二，车牌定位，就是在图片中确定出车牌的位置。

第三，字符识别，将提取出来的车牌图片用tesseract进行识别。

****

**2.2 图像分割算法设计**

使用sobel边缘检测算法可以提清晰的车牌轮廓

void chepai::sobel()

{

//sobel运算

//创建 grad\_x 和 grad\_y 矩阵

Mat grad\_x, grad\_y;

Mat abs\_grad\_x, abs\_grad\_y;

//求 X方向梯度

Sobel(gaussImage, grad\_x, CV\_16S, 1, 0, 3, 1, 1, BORDER\_DEFAULT);

convertScaleAbs(grad\_x, abs\_grad\_x);

/\* imshow("【效果图】 X方向Sobel", abs\_grad\_x);\*/

//求Y方向梯度

Sobel(gaussImage, grad\_y, CV\_16S, 0, 1, 3, 1, 1, BORDER\_DEFAULT);

convertScaleAbs(grad\_y, abs\_grad\_y);

/\* imshow("【效果图】Y方向Sobel", abs\_grad\_y);\*/

//合并梯度(近似)

addWeighted(abs\_grad\_x, 0.5, abs\_grad\_y, 0.5, 0, sobelImage);

/\* imshow("【效果图】整体方向Sobel", sobelImage);\*/

}

**2.3 图像增强算法设计**

高斯滤波过滤噪声

void chepai::gaussFilter()

{

//高斯滤波

GaussianBlur(srcImage, gaussImage, Size(3, 3), 0);

/\* imshow("【效果图】高斯滤波", gaussImage);\*/

## }

## 2.4车牌识别算法说明

调用tesseract文字识别API接口

void chepai::recognition()

{

TessBaseAPI orc;

const char \*langue = "eng+chi\_sim";

//char \*langue1 = "";

orc.Init(NULL, langue, OEM\_TESSERACT\_ONLY);

orc.SetVariable("tessedit\_write\_images", "1");

orc.SetPageSegMode(PSM\_SINGLE\_BLOCK);

orc.SetImage(reBinImage.data, reBinImage.cols, reBinImage.rows, reBinImage.elemSize(), reBinImage.cols);

char \*text = orc.GetUTF8Text();//.GetUTF8Text();

String s = U2G(text);

QString qstr2;

qstr2= QString::fromStdString(s);

ui.lineEdit->setText(qstr2);

}

2.5 用户接口设计

用户点击打开图片按钮打开图片，在label中显示原图，点击识别车牌按钮，在label\_2中显示轮廓图,lineedit中显示车牌号。

void retranslateUi(QMainWindow \*chepaiClass)

{

chepaiClass->setWindowTitle(QApplication::translate("chepaiClass", "chepai", nullptr));

label->setText(QApplication::translate("chepaiClass", "TextLabel", nullptr));

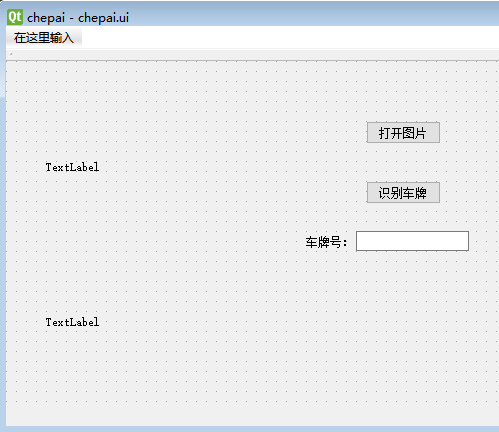
pushButton->setText(QApplication::translate("chepaiClass", "\346\211\223\345\274\200\345\233\276\347\211\207", nullptr));

pushButton\_2->setText(QApplication::translate("chepaiClass", "\350\257\206\345\210\253\350\275\246\347\211\214", nullptr));

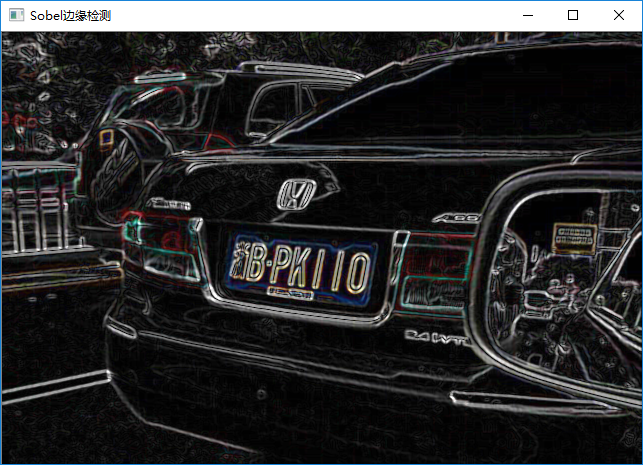
label\_2->setText(QApplication::translate("chepaiClass", "TextLabel", nullptr));

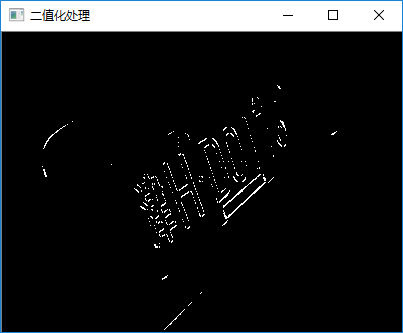
label\_3->setText(QApplication::translate("chepaiClass", "\350\275\246\347\211\214\345\217\267\357\274\232", nullptr));

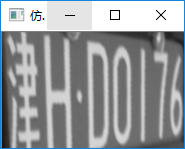
}

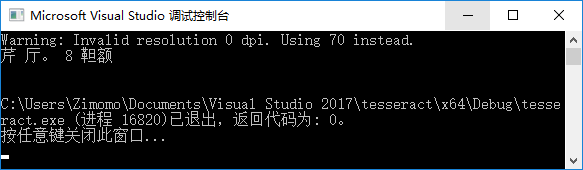


**三、系统实施及结果**

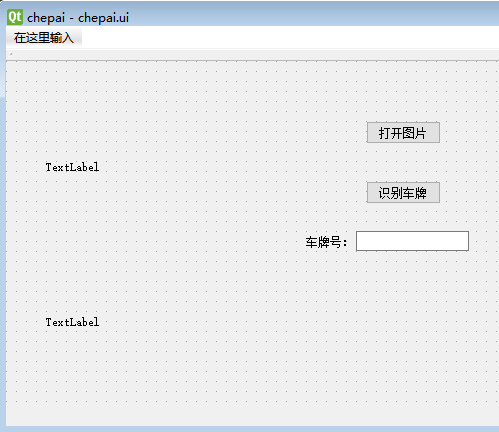


****

****



## **3.1系统操作说明**



用户点击打开图片按钮打开图片，在label中显示原图，点击识别车牌按钮，在label\_2中显示轮廓图,然后lineedit中显示车牌号。

1. **课程设计总结**

在window平台配置Opencv、Tesseract、qt是此次课程设计最麻烦的问题，两人共同协作，配置路径不同，重新配置时很容易出现新的错误。而且tesseract在window平台的配置操作非常繁琐，需要用到cmake等编译工具，编译费时。如果有机会，下次我没会选择类似python等便捷的平台。

其次是，编译了ui文件后，GeneratedFiles并不会自动添加ui\_\*\*\*.h文件，导致又出现了一系列的错误。还有就是一开始不明白怎么连接qt的信号与槽，不过才搜索了相关信息后，这个问题还是顺利解决了。