# 字符分割实验报告

## 概述

这个项目实现了分割精准定位过的车牌上的字符的功能，主要采用的轮廓检测的方法，在手工定位的数据集上有75%+的正确率

## 算法

这个项目的算法如下

1. 将图片读入，大小统一为标准大小
2. 对图片进行亮度提升操作
3. 对图片应用自适应阈值的二值化算法
4. 通过一定的算法去除边框和钉子
5. 对图片运行轮廓检测，确定轮廓矩形
6. 对矩形进行一系列预处理，对汉字进行特殊判定
7. 筛选、排序、返回

## 算法要点

### 亮度提升

采取了一个根据经验参数的亮度提升算法，主要目的是提高图片高度部分的亮度，为下一步的二值化操作做好准备。

### 二值化

采用了OpenCV自带的threshold函数，同时应用THRESH\_OTSU算法确定阈值。

### 去除边框

本项目实现了两个算法，最终采取了其中的算法二

算法一：从车牌中心在竖直方向向两边扫描，计算每次移动时，每行亮度总和的变化量，当变化量超过某一阈值时，确定这一行为上/下边界。

算法二：从车牌两边向下扫描，直到遇到满足条件（有7次突变，连续亮度和小于阈值，总亮度小于阈值，不超过某一坐标阈值）时，确定这一行为上/下边界。

由于算法一的阈值难以选择，故选择算法二来去除边框

### 轮廓检测

采用OpenCV自带的findContours，并确定包围的矩形

### 矩形预处理

车牌中的汉字很有可能是不联通的，所以会识别成多个矩形，由于汉字位置在最左边，首先确定第二个字符的位置，然后将第二个字符左面所有的矩形合并为大矩形，最后筛选一些明显错误的矩形

## 效果

本项目的效果采取人工检测(test.py)，经过多次抽样测试，分割完全准确的概率都在75%以上。

## 后记

由于上次了解了OpenCV和python的知识，这次就比较轻松了，总共用了4天中总计8小时的时间来完成这个项目，为了降低难度，选择了easy的样本。

首先第一直觉是用定位中用到的边缘检测和轮廓检测，试了几次后发现直接轮廓检测的效果更好，就沿着这个二值化->轮廓检测的思路走了下去。

遇到的第一个问题:和边框粘连，google了一下发现有基于先验知识的投影法可用，但是感觉不是非常优雅，便准备自己想想解决方案，想办法把边框切掉，首先看看直方图，感觉边框的直方图有突变，着手实现了一个通过判断突变大小来切割的算法，但是这个大小……阈值非常玄学，我觉得只能放弃，着手实现了算法二，也是比较常规的想法，最初只想到了7次突变，后来慢慢加上了连续长度、总亮度等trick，算是能把边框较好的去除。（其实说实话我不是非常满意）

然后是汉字的坑，首先尝试形态学变化，效果很糟糕，然后想了一下，不妨直接trick一下，把最左边的合并一下，效果还是不错的，就用了这个方案。

最后排个学筛选一下，本来想写个检测脚本，但是发现无从下手，不妨就人工检测一下，效果还不错，我觉得可以用来下一步识别字符的样本了。