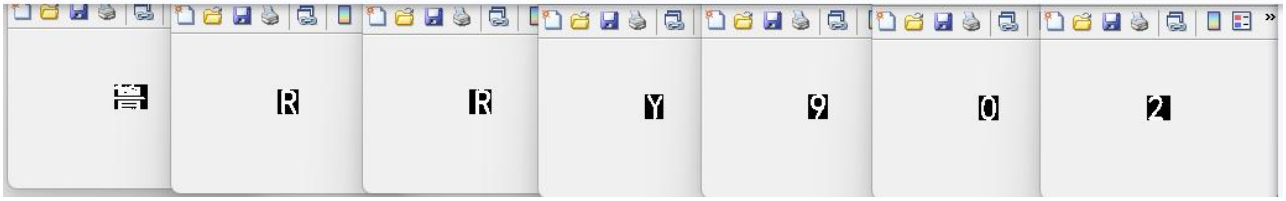
实验手册 8 图像分割——车牌识别

任务目标：完成小车的车牌定位与识别

实现效果：



实验流程：

1. 车牌定位

1）边缘检测（可调用 edge()，method 自行选择，然后反色成白底黑边）

2）开闭运算（对边缘检测图先开后闭）

图像形态学处理的一种手段，图像膨胀会扩大一幅图像的组成部分，而图像腐蚀会缩小一幅图像的组成部分。

开操作：先腐蚀，后膨胀，效果为平滑物体轮廓，断开较窄的窄颈并消除细的突出物

闭操作：先膨胀后腐蚀，有助于消除物体内部小黑点

用法及函数效果在实验所用函数处展示

完成后需要对过小的连通域进行删除（噪声）

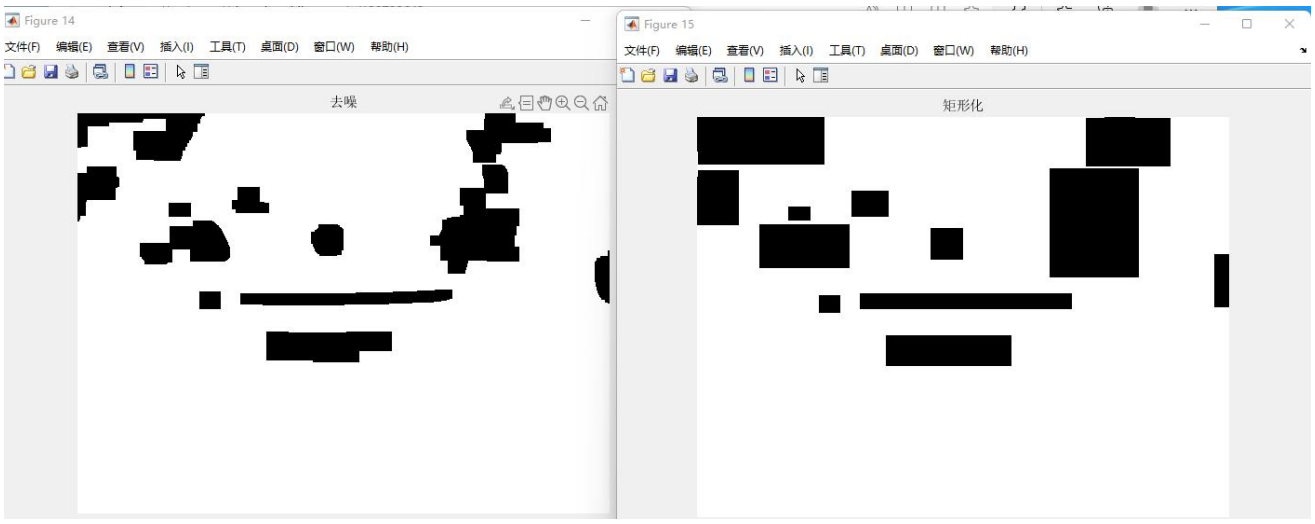
3）矩形化

对于连通区域进行矩形化，方便接下来选择车牌区域

从经过数学形态学运算得到的连通区域的图中可以看出，这些连通区域是不规则的，它们不利于后续的处理 （从候选区域中提取车牌区域）。所以，要使这些候选区域矩形化，即扩大这些候选区域的边缘，使其成为矩形区域。

其具体步骤如 下：①若 G（i，j）为黑点，而且它的 4－邻 域中有至少任意两像素为白点，则令其为白 点；②若 G（i，j） 为白点，而且它的 4 －邻域中至少 2 个像素为黑点，则令其为黑点。

该步骤要从（0，0）（m,0）（0,n）（m,n）四个起始点对图像做迭代循环，不用建新图，每次循环都是从上一次更新的图往下做



4）选择区域

对上列矩形区域进行筛选，对于车牌，我们利用如下约束进行筛选：

（1）矩形长宽比

（2）区域蓝色像素点所占比例（针对国内车牌为蓝底）

（3）候选区域垂直投影函数与其 y=平均值的这条直线交点的个数要在一定阈值内，count 在 15 到 45 之间。

其中条件 2，3 比较重要，3 用于排除颜色正确但样式不对的候选区域，3 排除蓝色以外的干扰项，垂直投影函数可调用 Verticalprojection.m 函数

2. 图像分割

对图片进行二值化，突出车牌数字，分别尝试下列两种方法并分析

1）基于全局的阈值分割（大津法）

详见课本 P114 或附录

2）基于局部的阈值分割（bernsen）

详见课本 p115 或附录

3. 字符分离

分割字符，利用栅栏法（即寻找字符间的空隙）分离字符，这里我们可以利用 y 方向上像素和为 0 来寻找分离点，也可以跳过此步做下一步识别

4. 车牌识别

本节自由发挥，可以使用第一次所做的模板匹配法，也可以用 python 机器学习来做

实验所需函数

J = imopen(I,SE) 使用结构元素 SE 对灰度或二值图像 I 执行形态学开运算。形态学开运算是先腐蚀后膨胀，这两种运算使用相同的结构元素。

imclose(I,SE) 同理
效果:(左闭右开)

SE = strel("rectangle",[m n]) 创建一个大小为 [m n] 的矩形结构元素

Test1 参数为 se 均为 [10，20], rato 为（2，8）区间，count 为（15，45）

Test2 参数为 se 开[5,5]，闭[10，20]，rato (1.5,2),count(5,15),blue 大于 0.9

[m,n] 我们一般选择 [10，20]，但具体参数还是视开闭操作后矩形是否覆盖车牌区域，或者有无跟其他区域联通在一起导致矩形面积过大的问题

Rgb2hsv() 将图像从 rgb 转到 hsv 域，用于检测蓝色区域占比

像素为蓝色的条件为:

```
if candidate_hsv(x1,y1,1)<0.667 && candidate_hsv(x1,y1,1)>0.5
    disp(candidate_hsv(x1,y,1));
    blue=blue+1;
```

edge(i，method) 边缘检测函数
~edge() 直接出反色边缘

对于矩形区域的选择，提供如下函数

[L,n] = bwlabel(BW,conn) 返回标签矩阵，其中 conn 指定连通性。L 为标注图像（矩阵），

n 为找到的连通区域数量，conn 取值为 4 或 8。

`stats = regionprops(L,properties)` 测量标注图像 L 中每个标注区域的一组属性。

`properties:` 'Area'区域中的实际像素数，以标量形式返回。

'BoundingBox'包含区域的最小外接框的位置和大小，以 1×(2*Q) 向量形式返回，其中 Q 是图像维度。前 Q 个元素是边界框最小边角的坐标。接下来的 Q 个元素是外接框沿每个维度的大小。例如，值为 [5.5 8.5 11 14] 的二维边界框表示外接框左上角 (x,y) 坐标为 (5.5, 8.5)，框的水平宽度为 11 个像素，垂直高度为 14 个像素。

"basic":计算 "Area"、"Centroid" 和 "BoundingBox" 测量值

附录：

保证目标的完整性和连通性。常见的局部阈值
普拉斯算法等。

1. 大津法

大津法是效果较好且应用比较广泛的一种全局阈值算法，其原理是把图像分为目标
和背景像素两类，以灰度值作为两类的特征属性，最佳阈值应该体现出最优的类别分
方差是对特征（这里为灰度值）分布离散性的一种度量，方差越大，说明构成图像的
的差别越大。因此引入类内方差、类间方差和总体方差，并定义 3 个等效的准则来衡
分离性能。好的全局阈值能使两个类别的类内方差小，类间方差大。大津法使用最大
差来获取最佳分割阈值。

该算法的具体步骤如下。

对尺寸为 $M\times N$ 的图像 $f(x,y)$，灰度值范围为[0, $m-1$]（通常为[0, 255]），$p(k)$表示灰
为 k 的概率，则有

$$p(k)=\frac{1}{MN}\sum_{f(i,j)=k}1 \tag{10-2}$$

则目标部分比例为

$$w_0(t)=\sum_{0\leqslant i\leqslant t}p(i)$$

目标部分像素数为

$$N_0=MN\sum_{0\leqslant i\leqslant t}p(i)$$

背景部分比例为

$$w_1(t)=\sum_{t\leqslant i\leqslant m-1}p(i)$$

背景部分像素数为

$$N_1=MN\sum_{t\leqslant i\leqslant m-1}p(i)$$

目标的灰度均值为

$$\mu_0(t)=\sum_{0\leqslant i\leqslant t}ip(i)/w_0(t)$$

背景的灰度均值为

$$\mu_1(t)=\sum_{t\leqslant i\leqslant m-1}ip(i)/w_1(t)$$

总的图像灰度均值为

$$\mu=w_0(t)\mu_0(t)+w_1(t)\mu_1(t)$$

那么，图像的最佳阈值 g 为

$$g=\max\left[w_0(t)(\mu_0(t)-\mu)^2+w_1(t)(\mu_1(t)-\mu)^2\right] \tag{10-3}$$

即令阈值在 0～$m-1$ 之间遍历，求使类间方差 $w_0(t)(\mu_0(t)-\mu)^2+w_1(t)(\mu_1(t)-\mu)^2$ 最大时的 g，即所求的最佳阈值。

2. 迭代法

迭代法也是常用的全局阈值算法，具体过程如下。

（1）选取初始阈值 T_0。通常可选择图像灰度值的中值作为初始阈值 T_0。

（2）利用阈值 T_i 把图像分割成两部分：R_1 和 R_2（目标和背景），分别计算两部分的灰度值，再将结果取平均，以获取一个新的阈值 T_{i+1}。

$$T_{i+1}=\frac{1}{2}\left[\frac{\sum_{k=0}^{T_i}h_k\times k}{\sum_{k=0}^{T_i}h_k}+\frac{\sum_{k=T_{i+1}}^{L-1}h_k\times k}{\sum_{k=T_{i+1}}^{L-1}h_k}\right] \tag{10-4}$$

式中，L 为灰度级的个数，h_k 是灰度值为 k 的像素点的个数。

（3）当新的阈值 T_{i+1} 与 T_i 的差小于某个给定值时，结束迭代。此时的 T_{i+1} 即图像分割的阈值。

3. Bernsen 算法

该算法将窗口内像素最大值与最小值的平均值作为当前像素点的阈值，计算公式如下：

$$T(x,y)=\frac{1}{2}\left(\max_{\substack{-W\leqslant m\leqslant W\\-W\leqslant n\leqslant W}}f(x+m,y+n)+\min_{\substack{-W\leqslant m\leqslant W\\-W\leqslant n\leqslant W}}f(x+m,y+n)\right) \tag{10-5}$$

式中，$T(x,y)$ 为 $f(x,y)$ 对应的阈值，此处选择的计算窗口是边长为 $2W$ 的正方形。根据各像素点的阈值对图像中的像素点逐个进行二值化。

4. Niblack 算法

光照非常不均匀时，对图像中的每个像素点计算其邻域的灰度均值 m 和标准差 s。图像中任意一点的阈值计算公式如下：

$$T=m+ks \tag{10-6}$$

式中，k 为自定义系数，k 越大，噪声去除效果越好，但目标区域也会变大。

先用高斯函数对