# 《计算机视觉（1）》实验报告

## 实验四 局部处理的边缘连接

| 实验小组成员<br>（学号+班级+姓名） | 分工及主要完成任务 | 成绩 |
| --- | --- | --- |
| 201800820149+18 数科+徐潇涵 | 二值图像生成、水平连接与垂直连接的边缘像素、前两幅图像的逻辑或、形态学细化、实验报告 | |
| 201800810253+18 数科+王棠 | 图像平滑处理、梯度幅度图像 | |
| | | |

山东大学

2021 年 3 月

完成《数字图像处理》P468页例10.10的编程实验，编程语言可以选择Matlab，C，C++，OpenCV，Python等。设计方案可参照教科书中的分析，也可以自行设计新的方案。

图(a)显示了一辆汽车尾部的图像。该例的目的是说明用前述算法来寻找大小适合车牌的矩形的应用。该矩形可以通过检测强的水平和垂直边缘构成。图(b)显示了梯度幅度图像M(x，y)，图(c)和(d)显示了该算法步骤3和步骤4的结果，其中，令TM等于最大梯度值的30%，A＝90°，TA＝45°，并填充了全部25个或更少像素的缝隙(约为图像宽度的5%)。

为检测车牌壳的全部拐角和汽车的后窗，要求使用一个较大范围的容许角度方向。图(e)是前两幅图像逻辑“或”(OR)操作的结果，图(f)是使用9.5.5节讨论的细化过程细化图(e)得到的。如图(f)所示，在图像中清楚地检测到了对应于车牌的矩形。如果汽车牌照的宽高比有与众不同的2∶1 的比例，所以利用这一事实从图像的所有矩形中简单地分离出牌照是一件简单的事情。

图(a)一幅大小为534×566的汽车后部图像；(b)梯度幅度图像；(c)水平连接的边缘像素；(d)垂直连接的边缘像素；(e)前两幅图像的逻辑“或”(OR)；(f)用形态学细化得到的最终结果

原始图像的电子版图像在 Images 文件夹中。实验报告写在如下空白处，页数不限。

# 实验报告

## 一、 实验内容

**1) 图像平滑处理，计算梯度幅度阵列和梯度角度阵列**

通过图像的后续显示结果可以看出，图像存在明显噪声，所以首先通过高斯滤波对图像进行平滑处理，并通过 sobel 算子计算出图像的梯度幅度阵列和梯度角度阵列。

**2) 通过阈值与带宽得到二值图像**

通过阈值对梯度幅度阵列进行限制，通过指定角度形成带宽对梯度角度阵列进行限制，选择符合条件的位置赋 1，完成边缘连接前的像素选择。

**3) 显示水平连接与垂直连接的边缘像素**

扫描二值图像的行，选择有足够的 1 值的行（排除噪声点），并在限制长度内的每一行中填充缝隙。进行图像的边缘连接。

**4) 水平连接与垂直连接的边缘像素图像的“或”处理**

通过对水平连接与垂直连接的边缘像素图像的“或”处理得到汽车完整的边缘像素图像。

**5) 图像的形态学细化**

对图像进行形态学细化，得到最终的结果图像。

## 二、 实验步骤

**1) 图像平滑处理，计算梯度幅度阵列和梯度角度阵列**

通过 OpenCV 中的高斯滤波函数与 Sobel 算子函数，进行图像平滑、梯度幅度阵列和梯度角度阵列的计算与梯度幅度图像的获取

```
car = cv2.imread("4.tif", 0)
car = cv2.GaussianBlur(car, (9,9), 0)
row, column = car.shape
car_f = np.copy(car)/255.0  # 像素值0-1之间

# sobel算子
sobelx = cv2.Sobel(car_f, -1, 1, 0, ksize=3) #只计算x方向
sobely = cv2.Sobel(car_f, -1, 0, 1, ksize=3) #只计算y方向
mag, ang = cv2.cartToPolar(sobelx, sobely, angleInDegrees=1)  # 得到梯度幅度和梯度角度阵列
cv2.imshow('M', mag)
cv2.imshow('x and y', np.hstack((sobelx, sobely)))
```

**2) 通过阈值与带宽得到二值图像**

教材中指定阈值为最大梯度值的 30%，通过对比调参等，实验发现将实验的最低阈值限制在最大梯度的值的 40%左右可以得到更好的图像处理效果，我们通过不同“带宽”分别将梯度角度限制在$\left(\frac{\pi}{4},\frac{3\pi}{4}\right),\left(\frac{5\pi}{4},\frac{7\pi}{4}\right)$与$\left(0,\frac{\pi}{4}\right),\left(\frac{3\pi}{4},\frac{5\pi}{4}\right),\left(\frac{7\pi}{4},2\pi\right)$中，获得水平连接边缘像素与垂直连接边缘像素。

**3) 显示水平连接与垂直连接的边缘像素**

在扫描二值图像的行，我们应选择有足够的 1 值的行，以排除噪声点，然后进行缝隙的填充，实现边缘连接。这里我们选择指定长度 K 值为 25。

```
# 行扫描，间隔k时，进行填充，填充值为1
def edge_connection(img, size, k):
    for i in range(size):
        Yi = np.where(img[i,:] > 0)
        if len(Yi[0]) > 11:  # 可调整
            for j in range(0, len(Yi[0]) - 1):
                if Yi[0][j + 1] - Yi[0][j] <= k:
                    img[i, Yi[0][j]:Yi[0][j + 1]] = 1
    return img


gx = edge_connection(g1, car_f.shape[0], k=25)
g2 = cv2.rotate(g2, 0)
gy = edge_connection(g2, car_f.shape[1], k=25)
gy = cv2.rotate(gy, 2)
```

**4) 水平连接与垂直连接的边缘像素图像的“或”处理**

在 python 中，我们通过‘+’来实现图像的或处理。

**5) 图像的形态学细化**

对图像进行形态学细化，得到最终的结果图像。

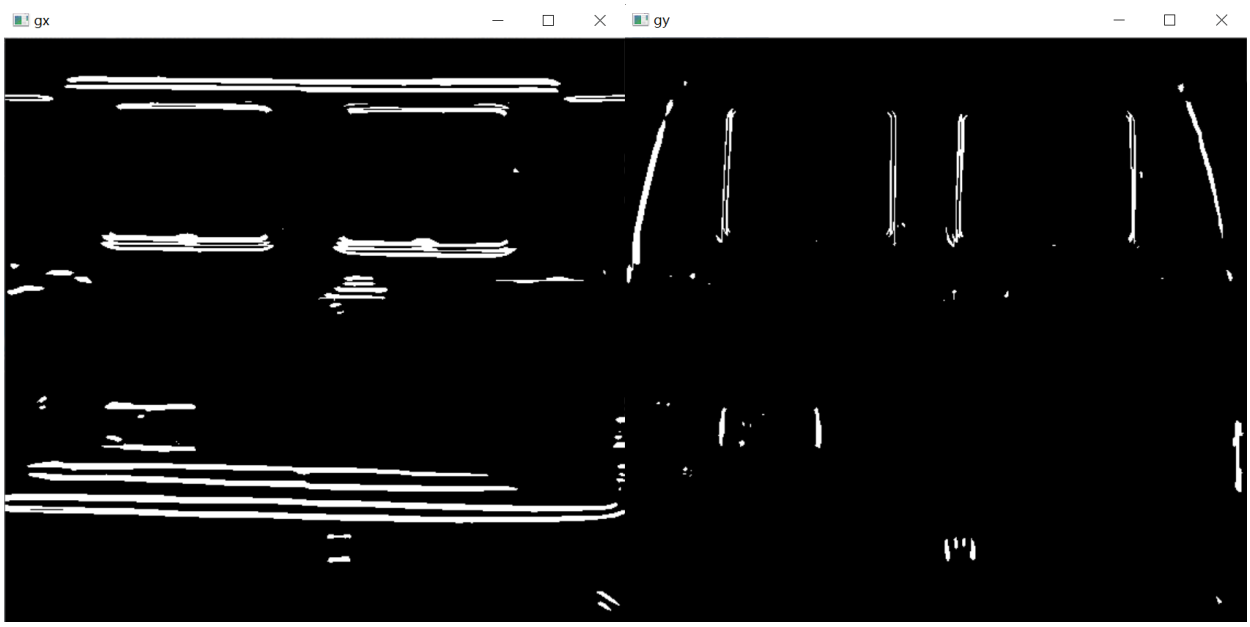
```
# 形态学细化
def refining(f):
    rows, cols = f.shape
    # 细化模板
    B1 = np.array([-1, -1, -1, 0, 1, 0, 1, 1, 1]).reshape(3, 3)
    B2 = np.array([0, -1, -1, 1, 1, -1, 1, 1, 0]).reshape(3, 3)
    B3 = np.array([1, 0, -1, 1, 1, -1, 1, 0, -1]).reshape(3, 3)
    B4 = np.array([1, 1, 0, 1, 1, -1, 0, -1, -1]).reshape(3, 3)
    B5 = np.array([1, 1, 1, 0, 1, 0, -1, -1, -1]).reshape(3, 3)
    B6 = np.array([0, 1, 1, -1, 1, 1, -1, -1, 0]).reshape(3, 3)
    B7 = np.array([-1, 0, 1, -1, 1, 1, -1, 0, 1]).reshape(3, 3)
    B8 = np.array([-1, -1, 0, -1, 1, 1, 0, 1, 1]).reshape(3, 3)
```

```
    maskList = [B1, B2, B3, B4, B5, B6, B7, B8]
    count = 0
    # skemask1 = np.array([1, 0, 1, 0, 1, 0, 1, 0, 1]).reshape(3, 3)
    while True:
        temp = f.copy
        for m in maskList:
            mas = []
            for i in range(1, rows - 1):
                for j in range(1, cols - 1):
                    if f[i, j] == 0:
                        continue
                    elif np.sum(m * f[i - 1:i + 2, j - 1:j + 2]) == 4:
                        # 击中时标记删除点
                        mas.append((i, j))
            for it in mas:
                x, y = it
                f[x, y] = 0
        if (temp == f).all:
            count += 1
        else:
            count = 0
        if count == 8:
            break
    return f
```

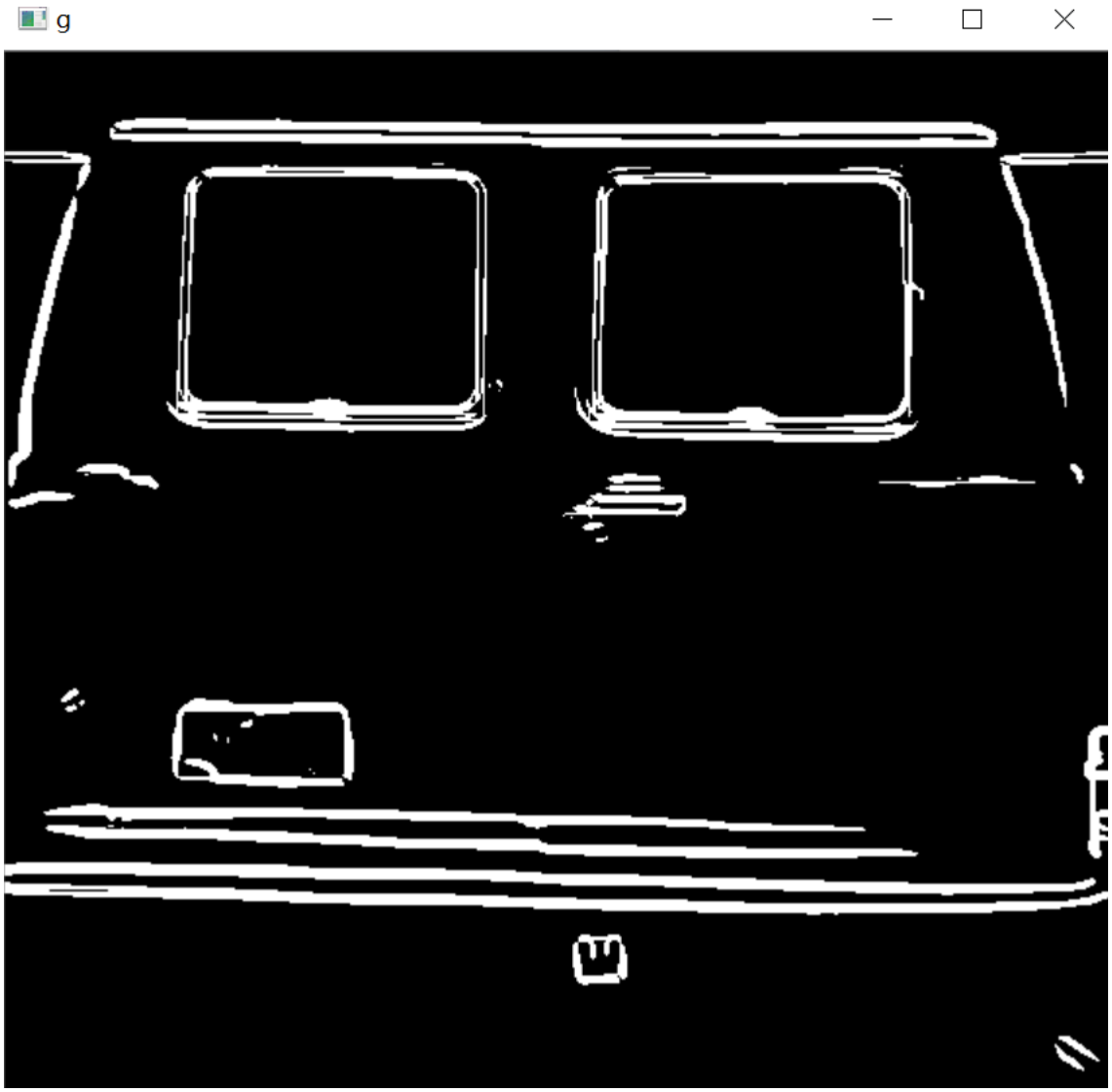
# 三、 实验结果

## 1) 图像梯度幅度图像

**2)水平连接与垂直连接的边缘像素图像**




**3) 水平连接与垂直连接的边缘像素图像的“或”处理**



**4) 图像的形态学细化**