课程名称：计算机图形图像技术试验

实验名称：边缘检测与车道线识别

**姓 名： 应一凡**

**专 业： 智能科学与技术**

**学 号： 21312274**

**实验一 边缘检测**

**一、实验内容及目的**

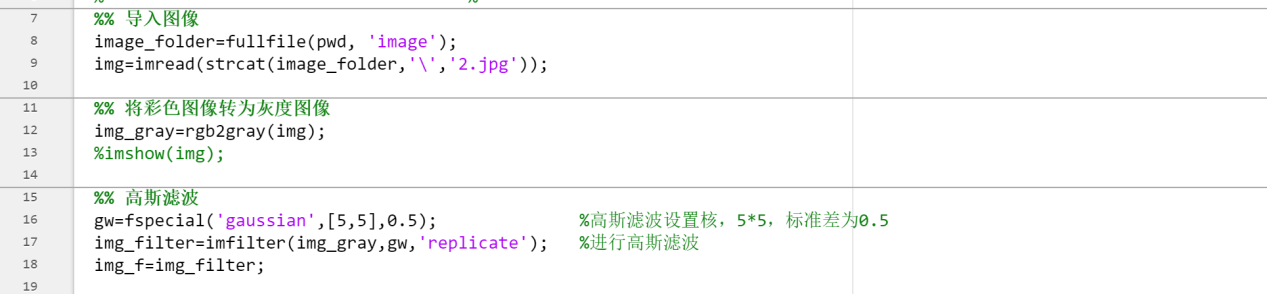
输出给定图像的边缘检测图像

1. **实验相关原理描述**
2. 高斯滤波
3. 利用 sobel 算子计算像素梯度
4. 对梯度图像进行非极大值抑制
5. 阈值滞后处理

5）孤立弱边缘抑制

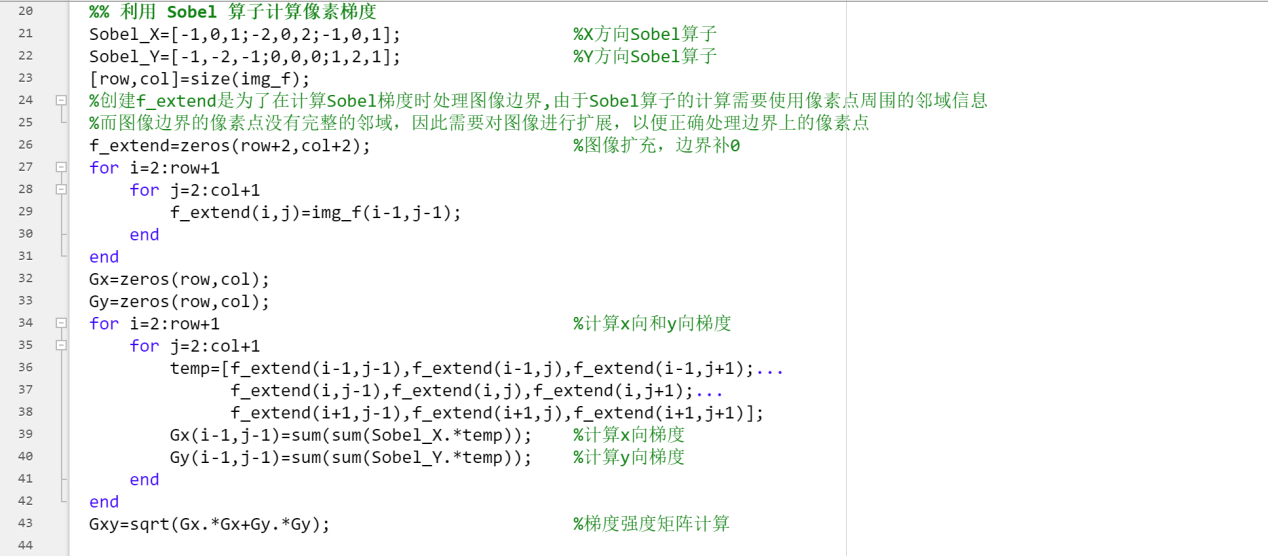
**三、实验过程**

1.图片的导入和预处理



使用fullfile函数获取当前目录下的'image'文件夹的完整路径，并将路径赋值给image\_folder变量。然后使用imread函数读取'image\_folder'目录下的'2.jpg'图像文件，并将读取到的图像数据存储在img变量中。用rgb2gray函数将彩色图像img转换为灰度图像，并将转换后的灰度图像数据存储在img\_gray变量中。使用fspecial函数创建一个高斯滤波器核，核大小为5x5，标准差为0.5，并将创建的滤波器核存储在gw变量中。然后使用imfilter函数对灰度图像img\_gray进行高斯滤波操作，滤波结果存储在img\_filter变量中

1. 利用sobel算子计算梯度

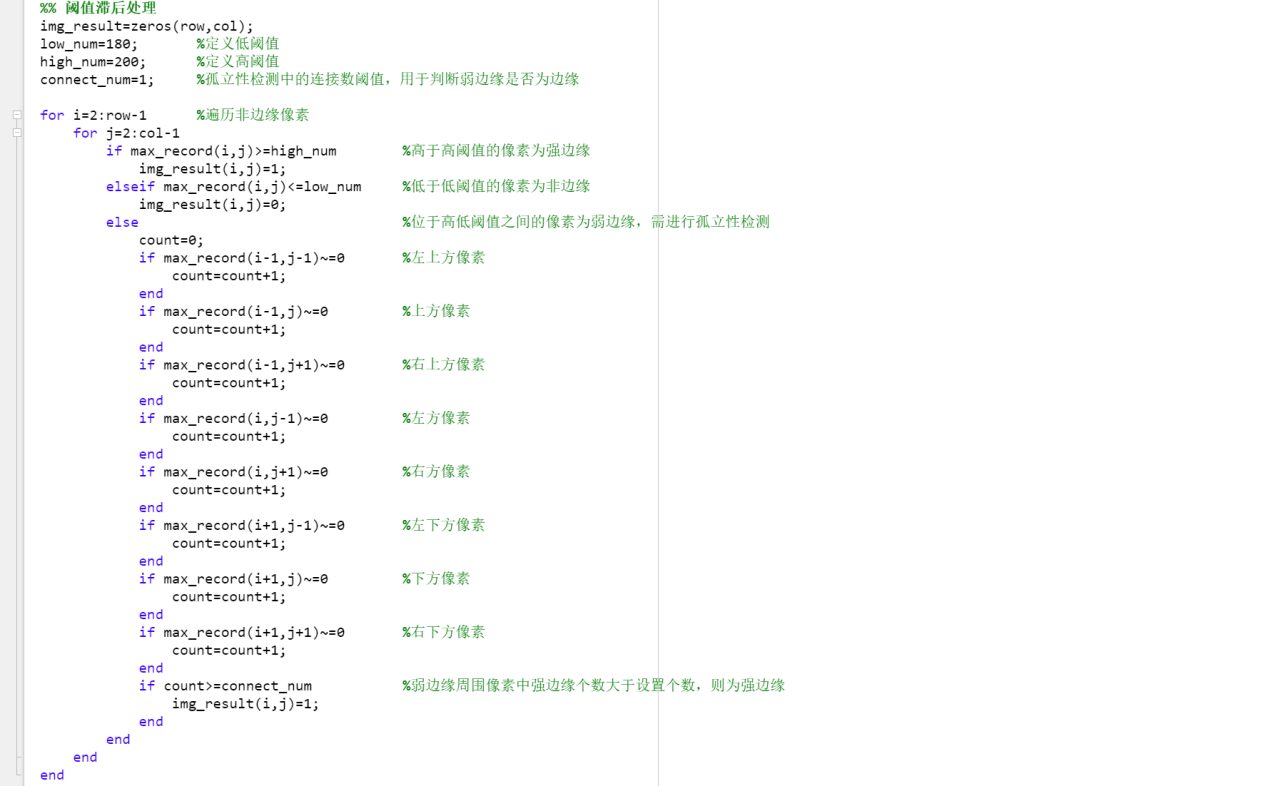
分别定义了X方向和Y方向的Sobel算子，存储在Sobel\_X和Sobel\_Y矩阵中。然后创建一个比原图像img\_f稍微大一圈的零矩阵f\_extend，用于在计算Sobel梯度时处理图像边界。通过嵌套的循环将img\_f中的像素值赋值给f\_extend中对应位置，实现图像的边界扩展。然后使用两层嵌套的循环遍历f\_extend中的像素点，对于每个像素点，提取其周围3x3邻域的像素值，存储在temp矩阵中。然后使用Sobel\_X和Sobel\_Y与temp进行元素对应相乘，并通过sum函数计算得到X方向和Y方向的梯度值，分别存储在Gx和Gy矩阵中。最后，使用sqrt函数计算每个像素点的梯度强度，并将结果存储在Gxy矩阵中。

1. 非极大值抑制



首先，根据输入的梯度图像 Gx 和 Gy，计算每个像素点的梯度方向，并将梯度方向分成五个区间，并用 index 数组记录每个像素点的梯度方向属于哪个区间（1-4）或无梯度（5）。然后，对于非边界处的像素点，根据其所属的梯度方向区间，进行插值计算，得到上方和下方的插值梯度强度，分别存储在 Gup 和 Gdown 数组中。最后，遍历所有像素点，将满足条件的极大值像素点记录在 max\_record 数组中，其中条件是当前像素点的梯度强度大于等于其上方和下方的插值梯度强度，不满足的不做处理，由于矩阵初始化为零矩阵，即实现抑制为0。

1. 阈值滞后处理



1）初始化一个与原图像大小相同的全零矩阵 img\_result，用于存储处理结果。定义三个阈值：low\_num 为低阈值，high\_num 为高阈值，connect\_num 为弱边缘连接数阈值。然后使用两层嵌套循环遍历非边缘像素，对每个像素点进行阈值判定：如果该像素点的像素值大于等于高阈值 high\_num，则将 img\_result 对应位置的像素值设为 1，表示为强边缘。如果该像素点的像素值小于等于低阈值 low\_num，则将 img\_result 对应位置的像素值设为 0，表示为非边缘。否则，即位于高低阈值之间的像素为弱边缘，需要进行孤立性检测。

2）孤立性检测：统计当前像素点周围 8 个邻近像素中强边缘的个数，如果大于等于设定的 connect\_num，则将 img\_result 对应位置的像素值设为 1，表示为强边缘。

3）循环结束后，img\_result 中存储了经过阈值滞后处理后的边缘检测结果图像。

1. 结果输出



在输出之前还用了系统函数作为比对，打印三个子图，分别是原图，自定义函数运行结果，系统函数运行结果

**四、实验结果**

****

中间的是自定义函数的运行结果，可以看到在不断修正参数（主要是阈值滞后处理的高低阈值）后，结果令人满意，好于系统自带的函数！

**五、总结**

通过这次实验，我深入了解了图像处理中的一些基本操作，包括阈值处理、高斯滤波、Sobel算子计算像素梯度和非极大值抑制等，对图像处理流程有了更加清晰的认识。同时，实验还让我了解了MATLAB中相关的图像处理函数的使用方法，并学会了如何在MATLAB中实现图像边缘检测的基本过程。通过对实验代码的调试和修改，我对图像处理中的一些概念和算法有了更深入的理解。

**实验二 车道线识别**

1. **实验内容及目的**

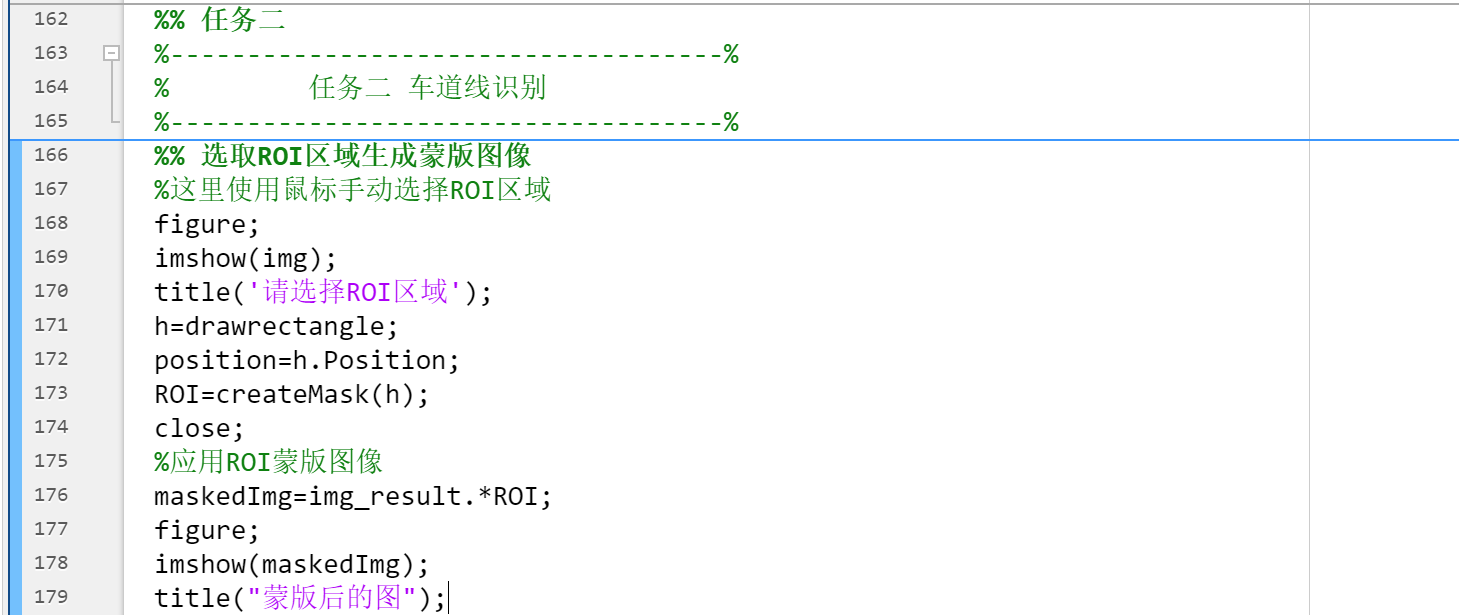
完成车道线识别

1. **实验相关原理描述**
2. 选取 ROI 区域生成蒙版图像
3. 霍夫（hough）变换识别直线边缘

3）根据车道线先验知识筛选直线

1. **实验过程**

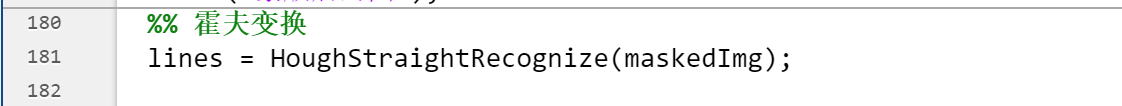
1.蒙版操作



由于上个实验已经有了边缘检测后的成品图，所以直接拿来用即可。

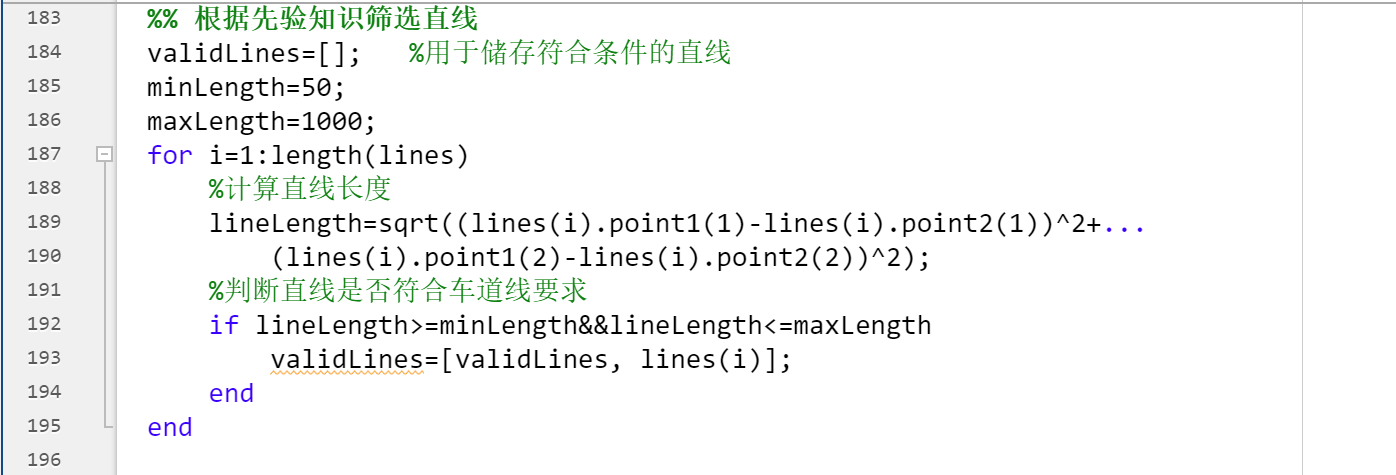
h=drawrectangle调用drawrectangle函数，允许用户在图像上手动绘制一个矩形框作为ROI，并将返回的drawrectangle对象赋值给变量h。然后获取用户绘制的矩形框的位置信息，并将其保存到变量position中。接着调用createMask函数，根据用户绘制的矩形框生成一个蒙版图像，并将其保存到变量ROI中。再将原始图像img\_result与生成的蒙版图像ROI进行逐像素的乘法操作，得到蒙版后的图像，并将其保存到变量maskedImg中。顺便打印出来看蒙版效果。

1. 霍夫变换



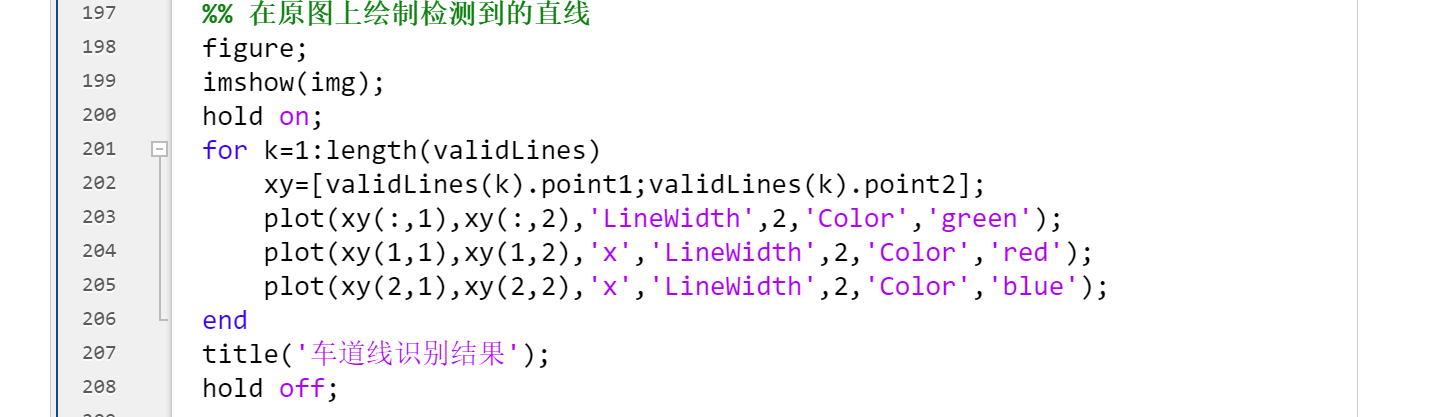
直接调用ppt上所给的函数

1. 根据先验知识筛选直线



由于斜率数值较难控制，这里只采用了长度限制。创建一个空的数组 validLines，用于储存符合条件的直线。定义了两个参数 minLength 和 maxLength，分别表示直线的最小长度和最大长度的阈值。进行一个 for 循环，遍历lines，在循环体内，计算两点之间的欧式距离，判断是否在设定的最小长度和最大长度的范围内，如果是则添加到 validLines 数组中。

1. 绘制直线



利用plot绘制所有validLines中的直线和它们各自的两个顶点

1. **实验结果**
2. 蒙版后的图像



可以观察到蒙版效果很好，上部分图像为全黑

1. 车道线的识别



在不断调整参数后，车道线的识别终于达到预期，可以观察到两条显著的车道线均被识别出来，其余较短或者不符合规范的线均被剔除

1. **总结**

1.学会了更多的图像处理技能，例如ROI区域生成蒙版图像、霍夫变换等

2.懂得了先验知识的应用，在实际问题中运用先验知识来优化算法或筛选结果，能够提高解决问题的准确性和效率。

3.学会了一个实际问题的解决方法，能够将图像处理技术运用于生活