实训四：图像边缘检测与锐化

作者：杨仕龙

1 实验的目的

1. 了解图像锐化的目的和原理；
2. 掌握常用的一阶差分：Sobel、Prewitt、LoG 算子和二阶差分：Laplacian 算子的理论和使用；
3. 掌握图像增强的综合处理方法。

2 实验原理

1. **图像锐化的目的和原理：**在图像的识别中常需要突出边缘和轮廓信息。图像锐化就是增强图像的边缘或轮廓。边缘和轮廓常常位于图像中灰度突变的地方，图像平滑通过平均（类似积分）过程使得图像边缘模糊，图像锐化则通过微分而使图像边缘突出、清晰。

2. **微分算子**具有突出灰度变化的作用，对图像运用微分算子，灰度变化较大的点处算得的值比较高，因此可将这些微分值作为相应点的边界强度，通过设置门限的方法，提取边界点集。一阶微分 $\frac{\partial f}{\partial x}$ 与 $\frac{\partial f}{\partial y}$ 是最简单的微分算子，它们分别求出了灰度在 x 和 y 方向上的变化率，而方向 α 上的灰度变化率可以用下面式子计算：

$$\frac{\partial f}{\partial \alpha}=\frac{\partial f}{\partial x}\cos\alpha+\frac{\partial f}{\partial y}\sin\alpha=G\cdot(\cos\alpha i+\sin\alpha j) \tag{1}$$

对于数字图像，应该采用差分运算代替求导，相对应的一阶差分为：

$$\begin{cases}\Delta_x f(i,j)=f(i,j)-f(i-1,j)\\ \Delta_y f(i,j)=f(i,j)-f(i,j-1)\end{cases} \tag{2}$$

方向差分为：$\Delta_\alpha f(i,j)=\Delta_x f(i,j)\cos\alpha+\Delta_y f(i,j)\sin\alpha$。函数 f 在某点的方向导数取得最大值的方向是 $\alpha=\tan^{-1}\left(\frac{\partial f}{\partial x}\Big/\frac{\partial f}{\partial y}\right)$。方向导数的最大值是 $|G|=\left[\left(\frac{\partial f}{\partial x}\right)^2+\left(\frac{\partial f}{\partial y}\right)^2\right]^{\frac{1}{2}}$ 称为梯度模。利用梯度模算子来检测边缘是一种很好的方法，它不仅具有位移不变性，还具有各向同性。为了运算简便，实际中采用梯度模的近似形式，如：$|\Delta_x f(i,j)|+|\Delta_y f(i,j)|$、$\max(|\Delta_x f(i,j)|,|\Delta_y f(i,j)|)$ 及 $\max(|f(i,j)-f(m,n)|)$。

3. **边缘检测**

   (a) Roberts 算子的表达式为：

$$\nabla_x=\begin{bmatrix}-1&0\\0&1\end{bmatrix},\nabla_y=\begin{bmatrix}0&-1\\1&0\end{bmatrix} \tag{3}$$

   Roberts 算子在水平方向和垂直方向的计算公式如下：

$$\begin{cases}\nabla_x(i,j)=f(i+i,j+1)-f(i,j)\\ \nabla_y(i,j)=f(i,j+1)-f(i+1,j)\end{cases} \tag{4}$$

   最后，我们需要对两个方向的计算结果求取算术平均：$S=\sqrt{\nabla_x(i,j)^2+\nabla_y(i,j)^2}$。

(b) Sobel 算子的表达式为：

$$\nabla_x = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}, \nabla_y = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \tag{5}$$

其中，∇_x 为 x 方向算子，∇_y 为 y 方向算子。由于 Sobel 算子是滤波算子的形式，用于提取边缘。我们可以利用快速卷积函数，简单有效，因此应用很广泛。

(c) 拉普拉斯算子的表达式如下：

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{6}$$

其中：

$$\begin{cases} \frac{\partial^2 f}{\partial x^2} = f(x+1, y) - 2f(x, y) + f(x-1, y) \\ \frac{\partial^2 f}{\partial y^2} = f(x, y+1) - 2f(x, y) + f(x, y-1) \end{cases} \tag{7}$$

将方程 (7) 代入到 (6) 可得：$\nabla^2 f(x, y) = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$，写成矩阵形式的算子为 K_1。

$$K_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, K_2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{8}$$

(d) **补充知识：**拉普拉斯-高斯算法（LoG）是一种二阶边缘检测方法。它通过寻找图像灰度值中二阶微分中的过零点（Zero Crossing）来检测边缘点。其原理为：灰度级变形成的边缘经过微分算子形成一个单峰函数，峰值位置对应边缘点；对单峰函数进行微分，则峰值处的微分值为 0，峰值两侧符号相反，而原先的极值点对应于二阶微分中的过零点，通过检测过零点即可将图像的边缘提取出来。

4. **OpenCV 中的边缘检测**

(a) Canny 算子

`edges = cv.Canny(image,threshold1,threshold2[,edges[,apertureSize[,L2gradient]]]);`

其中：低于阈值 1（threshold1）的像素点会被认为不是边缘；高于阈值 2（threshold2）的像素点会被认为是边缘；在阈值 1 和阈值 2 之间的像素点，若与第 2 步得到的边缘像素点相邻，则被认为是边缘，否则被认为不是边缘。

(b) Sobel 算子

`edges = cv2.Sobel(src,ddepth,dx,dy[,dst[,ksize[,scale[, delta[, borderType]]]]]);`

(c) Roberts 算子

`filter2D(src, ddepth, kernel, dst=None, anchor=None, delta=None, borderType=None);`

```
# Roberts 算子关键代码
krnl_x = np.array([[-1, 0], [0, 1]], dtype=int)
krnl_y = np.array([[0, -1], [1, 0]], dtype=int)
x = cv.filter2D(grayImage, cv.CV_16S, krnl_x)
y = cv.filter2D(grayImage, cv.CV_16S, krnl_y)

# 转 uint8 并融合图像
absX = cv.convertScaleAbs(x)
absY = cv.convertScaleAbs(y)
Roberts = cv.addWeighted(absX, 0.5, absY, 0.5, 0)
```

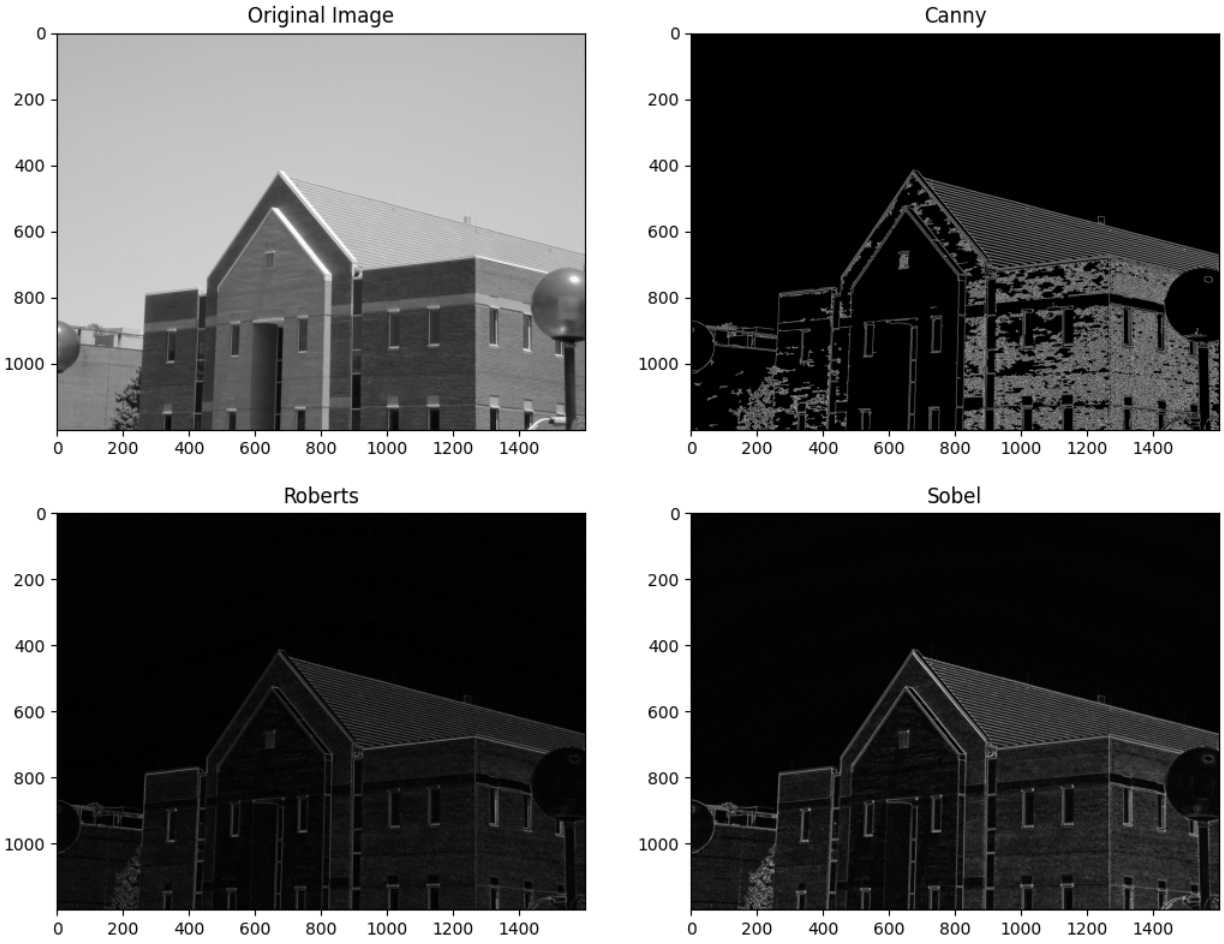

图 1: 边缘检测

3 实验内容

1. 读取 Images 文件夹中的一幅图像，并显示。
2. 分别用 Roberts、Sobel 和 Canny 算子对图像进行边缘检测。比较三种算子处理的结果。若一切正常，你将看到如图1所示结果。
3. 用不同方向（“水平”、“垂直”、“水平和垂直”）的 Sobel 算子对图像进行边缘检测。比较三种情况的结果。
4. 用拉普拉斯算子对月亮表面图像进行去模糊滤波。（请参考 Roberts 算子的实现代码，使用公式 (8) 中的 K_1 完成滤波。）
5. 按照一定的的处理顺序，综合运用多种图像增强方法对图像 bone.tif 进行处理。