作业1：图像滤波与边缘检测

任务一：基本处理

- 数据：lena灰度图和彩色图
- 要求：分别对lena灰度图和彩色图进行不同尺度和不同窗口大小的滤波和边缘检测

  其中：

  > 滤波器：高斯滤波核
  > 边缘检测子：高斯拉普拉斯检测子
  > 窗口大小：从*33到11*11，以2为间隔
  > 尺度因子：从1像素到7像素，以2为间隔

1. 实验原理

1.1 滤波：

滤波是图像处理中常用的操作，它通过卷积操作对图像进行平滑或增强边缘等效果。高斯滤波核是一种常见的平滑滤波器，通过对图像进行高斯卷积操作，可以有效地降低噪声，平滑图像。

1.2 边缘检测：

边缘检测是图像处理中的重要任务，可以帮助我们找到图像中的目标边界。高斯拉普拉斯检测子结合了高斯滤波和拉普拉斯算子，对图像进行边缘检测。它能够在边缘处产生较强的响应，帮助我们定位图像中的边缘。

1.3 OpenCV库

使用OpenCV库来进行图像处理，包括滤波和边缘检测：

- **高斯滤波（Gaussian Blur）：**
  - `cv2.GaussianBlur(src, ksize, sigmaX[, dst[, sigmaY[, borderType]]])`
  - 通过对图像应用高斯滤波，可以减小噪声，平滑图像。
  - 参数说明：
    - `src`：输入图像。
    - `ksize`：滤波核的大小，通常是正奇数。
    - `sigmaX`：X轴方向上的标准差。
    - `sigmaY`：Y轴方向上的标准差（如果为零，则与sigmaX相同）。
- **Laplacian算子：**
  - `cv2.Laplacian(src, ddepth[, dst[, ksize[, scale[, delta[, borderType]]]]])`
  - 使用Laplacian算子进行边缘检测，强调图像中的高频信息。
  - 参数说明：
    - src： 输入图像（单通道图像，如灰度图）。
    - ddepth： 输出图像的深度（图像数据类型），通常设置为cv2.CV_64F。
    - dst： 输出图像，可选参数。如果不提供，则函数会直接在原始图像上进行操作。
    - ksize： Laplacian核的大小。可选参数，默认值为3。通常使用3或者5。
    - scale： 可选参数，表示应用于Laplacian结果的比例因子。默认值为1。
    - delta： 可选参数，表示要加到结果中的偏移值。默认值为0。

- borderType： 可选参数，表示图像边界处理的方式。默认值为cv2.BORDER_DEFAULT。

2. 实验结果：

2.1 滤波结果：

通过对lena灰度图和彩色图进行不同尺度和窗口大小的高斯滤波，观察到图像的平滑效果。随着窗口大小的增加，图像平滑度增加，但同时可能丧失一些细节。尺度因子sigma为1时，窗口大小改变对图像平滑程度的改变并不明显，而在sigma为11时改变很明显。

2.2 边缘检测结果：

使用高斯拉普拉斯检测子进行边缘检测，观察到在图像边缘产生了强烈的响应。边缘的清晰度和精确性受到窗口大小和尺度的影响，不同参数下可以得到不同程度的边缘强调效果。窗口大小越大，图像变得更加平滑、模糊，并且能检测到的边缘变少。小尺度因子如sigma值小，使得图像中的一些细节边缘变得更加明显，能观察到头发和帽子的细微线条。

任务二：Canny检测子

- 数据：lena灰度图
- 要求：调用函数实现lena灰度图的Canny边缘检测，学习Canny边缘检测源代码

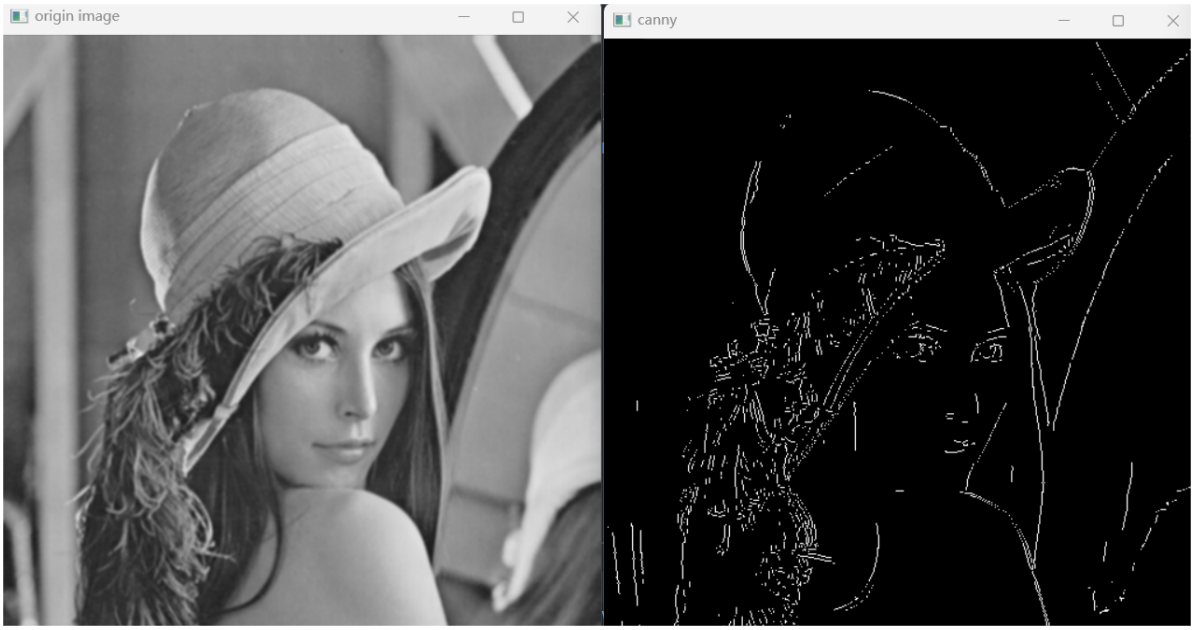
1、实验原理

Canny边缘检测是一种广泛使用的边缘检测算法，它主要包括以下步骤：

- **高斯滤波：** 使用高斯滤波器对图像进行平滑，以降低噪声的影响。
- **计算梯度：** 使用Sobel算子计算图像的梯度，找到图像中的强度变化。
- **非极大值抑制：** 在梯度方向上，对每个像素进行检查，保留梯度变化最大的像素，抑制其他像素。
- **双阈值处理：** 设置两个阈值，将梯度图像分为强边缘、弱边缘和非边缘三部分。
- **边缘跟踪：** 通过连接强边缘像素和其周围的弱边缘像素，形成完整的边缘。

2、实验结果

结果如下图所示：



任务三（拓展-可选）：

- 数据集：BSDS500数据集 https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html
- 要求：使用Canny边缘检测或某一种边缘检测算法对测试集进行边缘检测，与Ground-Truth进行对比，计算出精确率和召回率
- 具体：结果可参照网页中的结果图

  下载数据集，对其中images文件夹的图片进行边缘检测，并且与GT_convert_0文件夹中的图片进行对比，精确率和召回率如下图所示：

```
precision 0.058643333099045805
recall 0.218222715880217724
```