实验二：边缘检测与多尺度 Retinex 算法

实验目的：

1. 比较 Sobel 算子和 8 邻域拉普拉斯算子在边缘检测方面的差异性。
2. 实现多尺度 Retinex 算法，观察其对图像增强的效果。

实验内容：

1. 边缘检测算法比较

1.1 算法简介

- **Sobel 算子**：通过计算图像的水平和垂直梯度来检测边缘。其卷积核为：

$$\text{Sobel}_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad \text{Sobel}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

通过对图像进行卷积，可以提取出边缘信息。

- **8 邻域拉普拉斯算子**：利用二阶导数，能够检测到所有方向的边缘。其卷积核为：

$$\text{Laplacian} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

1.2 实验步骤

(1)、图像准备：

- 选择一幅包含明显边缘的测试图像（test.jpg）.
- 将其转换为灰度图.

(2)、实现 Sobel 算子：

- 使用定义的 Sobel 卷积核对图像进行卷积，分别计算水平和垂直梯度 (如下为 matlab 代码示例)：

```
grad_x = imfilter(double(gray_img), sobel_x);
grad_y = imfilter(double(gray_img), sobel_y);
```

- 计算幅值。根据设定的阈值提取边缘，并将其设置为 255，而其它非边缘部分则设置为一个较小的灰度值。

(3)、实现 8 邻域拉普拉斯算子：

- 对图像进行卷积，提取边缘信息 (如下为 matlab 代码示例)：

```
laplacian_img = imfilter(double(gray_img), laplacian);
```

- 根据阈值生成高亮图像。

(4)、可视化比较

1.3 结果分析：通过观察结果图，分析两种算法的优缺点以及适用场景。例如，Sobel 算子适用于具有清晰边缘的场景，而拉普拉斯算子则适用于细节丰富的场景等。

2. 多尺度 Retinex 算法实现

2.1 算法简介

- 多尺度 Retinex（MSR）算法通过结合不同尺度的高斯滤波结果，提高图像的对比度和细节表现。其公式为：

$$MSR(x,y) = \frac{1}{N}\sum_{i=1}^{N}\left(\log(I(x,y)+\epsilon) - \log(G_{\sigma_i} * I(x,y)+\epsilon)\right)$$

- 其中 I(x, y)是输入图像的像素值，G_σi 是标准差为 σi 的高斯滤波器，N 是使用的高斯滤波器的数量，ϵ 是防止对数计算中零值的小常数。

2.2 实验步骤

(1)、图像准备：

选择待增强的图像 (test.jpg)。

(2)、实现多尺度 Retinex 算法：

- 定义高斯滤波器的标准差，通常选择多个尺度（一般选择 3 个不同的尺幅为最佳）。
- 对每个尺度应用高斯滤波，提取不同频率的图像特征 (如下为 matlab 代码示例)。

```
G = fspecial('gaussian', [5 5], sigmas(i));   % 生成第 i 个高斯滤波器
filtered_img = imfilter(double(gray_img), G);
```

(3)、展示结果：对输出结果进行归一化处理，确保图像的亮度在合理范围内

(4)、结果分析：

- 与原始图像相比，增强后的图像在对比度提升及细节保留等方面的效果如何？
- 讨论多尺度 Retinex 算法在实际应用中的有效性和局限性。

实验总结

注意：1. 作业提交时间为 10 月 28 日晚 20:00 前

2. 作业提交晚于上述时间一天，扣 5%的分数、晚于两天扣 10%的分数、晚于三天及以上，本次实验课成绩为无效！