hw2

3.1 数字图像为离散的直方图，而均衡化后，由于一对一或多对一的映射关系，原图像的几个灰度级可能变为一个灰度级，所以不能产生完全平坦直方图。

3.2 原累积直方图 S_k 取整

$$S_k = int[(L-1)S_k + 0.5]$$

$k \to S_k$

增强 $(k \to S_k)$

对处理后的 S_k'

$$S_k' = int((L-1)S_k' + 0.5) = S_k$$

增强处理不变。

3.3.

1	原始灰度等级	0	1	2	3	4	5	6	7
2	原始直方图	0.174	0.088	0.086	0.08	0.068	0.058	0.062	0.280
3	原始累积直方图	0.174	0.262	0.348	0.428	0.496	0.554	0.616	1
4	规定直方图	0	0.4	0	0	0.2	0	0	0.4
5	规定累积直方图	0	0.4	0.4	0.4	0.6	0.6	0.6	1
6S	SML	1	1	1	1	1	4	4	7
7S	映射关系	0,1,2,3,4→1					5,6→4		7→7
8S	变换直方图	0	0.496	0	0	0.12	0	0	0.384

6G GML 1 1 1 1 4 4 4 7

7G 映射关系 0,1,2,3→1 4,5,6→4 7→7

8G 变换直方图 0 0.428 0 0 0.188 0 0 0.384

3.4

数字图像处理hw2 3.4

牛庆源 PB21111733

1. **定义滤波器大小**： 首先，确定滤波器的大小n。通常n是一个奇数，这样可以确保滤波器有一个中心点。
2. **遍历图像**： 遍历图像的每个像素位置。对于图像中的每个位置，我们将应用中值滤波器。
3. **获取邻域像素**： 对于当前像素位置，获取其周围n*n邻域的像素值。确保在图像边界处理时不越界。
4. **计算中值**： 对于获取的邻域像素，将它们排序，并找到其中值。
5. **更新像素值**： 将当前像素位置的值替换为计算得到的中值。

- Python代码如下:

```python
import numpy as np
import cv2

def median_filter(img, kernel_size):
    # 图像的行数和列数
    rows, cols = img.shape[0], img.shape[1]
    # 用于存储滤波后的图像
    filtered_img = np.zeros_like(img)

    # 遍历图像的每个像素
    for i in range(rows):
        for j in range(cols):
            # 计算滤波器的边界
            top = max(0, i - kernel_size // 2)
            bottom = min(rows, i + kernel_size // 2 + 1)
            left = max(0, j - kernel_size // 2)
            right = min(cols, j + kernel_size // 2 + 1)

            # 提取邻域像素
            neighbors = img[top:bottom, left:right]

            # 计算中值
            median_value = np.median(neighbors)

            # 更新像素值
            filtered_img[i, j] = median_value

    return filtered_img

# 读取图像
img = cv2.imread('input_image.jpg', 0)  # 以灰度模式读取图像
# 定义滤波器大小
kernel_size = 3
# 应用中值滤波器
filtered_img = median_filter(img, kernel_size)
# 显示原始图像和滤波后的图像
cv2.imshow('Original Image', img)
cv2.imshow('Median Filtered Image', filtered_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```