《数字图像处理》实验报告

姓名： 汪雨卿 学号：19120191

实验九

一. 任务 1

自己编程实现书本 10.3.2（基本的全局阈值处理）和 10.3.3（最优全局阈值处理）中提到的两种分割方法，对 rice.tif，finger.tif 和 poly.tif 进行分割，并对比结果。

a) 核心代码：

1. 基本的全局阈值处理

i. 新阈值计算

```
# 对图片计算新的阈值
def Thred(img, T):
    height, weight = img.shape[:2]
    A1 = A2 = 0
    B1 = B2 = 0
    for i in range(height):
        for j in range(weight):
            if img[i, j] > T:
                A1 += img[i, j]
                B1 += 1
            else:
                A2 += img[i, j]
                B2 += 1
    m1 = int(A1 / B1)
    m2 = int(A2 / B2)  # m1，m2计算两组像素均值
    T0 = int((m1 + m2) / 2)  # 据公式计算新的阈值
    return T0
```

ii. 进行图像分割，T0 为界限

```
def DoChange(img, T):
    height, weight = img.shape[:2]
    img_new = np.zeros((height, weight), np.uint8)
    T0 = T
    T1 = Thred(img, T0)
    # 迭代次数自定义，根据具体情况跳转设置
    for k in range(100):
        # 若新阈值减旧阈值差值为零，则为二值图最佳阈值
        if abs(T1 - T0) == 0:
            for i in range(height):
                for j in range(weight):
                    if img[i, j] > T1:
                        img_new[i, j] = 255
                    else:
                        img_new[i, j] = 0
            break
        else:
            T2 = Thred(img, T1)
            T0 = T1
            # 变量转换，保证if条件为新阈值减旧阈值
            T1 = T2
    return img_new
```

1. 最优全局阈值处理

计算阈值

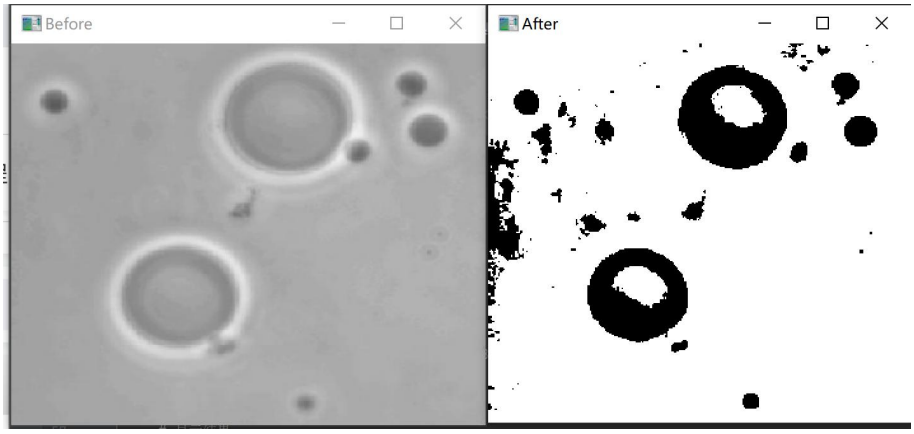
```
# 计算整体均值
meanTotal = np.sum(np.dot(h, np.array([n for n in range(256)])))
meanTotal = meanTotal / np.sum(np.array([n for n in range(256)]))
Mscore = 0;
gi = []
for i in range(1, 255):
    # 分割图像的均值只需要将直方图的高度乘以x坐标求和，再除以x坐标之和
    mean_y = np.sum(np.dot(h[0:i], np.array([n for n in range(i)])))
    mean_y = mean_y / np.sum(h[0:i])
    mean_x = np.sum(np.dot(h[i:256], np.array([n for n in range(i, 256)])))
    mean_x = mean_x / np.sum(h[i:256])
    # 公式计算
    score = sum(h[0:i]) * ((meanTotal - mean_y) ** 2) + sum(h[i:256]) * ((meanTotal - mean_x) ** 2)
```

b) 实验结果截图

1. 基本的全局阈值处理结果

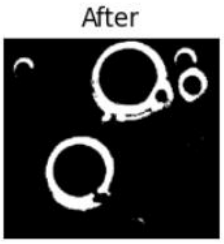
Finger.tif

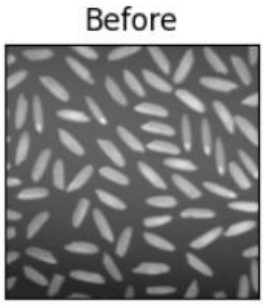
Poly.tif

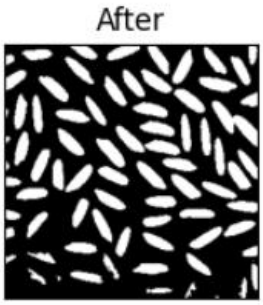


Rice.tiff

1. 最优全局阈值处理结果







四、实验小结

本实验基于课程中介绍的两中对于图像进行分割优化的实现。通过对于图像的灰度进程分割，对于一些具有某些特点的图片，尤其是二值的图像可以较好的突出想要的图像区块，便于一些图像检测的实现。通过对比大津法和基本的做法，可以发现大津法实现的噪声会更少，且轮廓更为圆滑。对于 poly.tif 这幅图来看，大津法生成的图片边缘区域更小，图像边缘更清晰，而基本做法则有些欠缺。因此对于不同的图片选择适用的方法也至关重要。