
# 数字图像处理实验报告

## 项目名称：基于直方图的图像空域变换和增强

姓名：刘朔江

班级：自动化钱 91

学号：2196113328

### 摘　要

图像的直方图（Histogram）横坐标是图像离散的灰度值，纵坐标是每个灰度值对应的像素点数目，它能够形象的显示出图像的灰度分布，通过对直方图的变换，可以平衡图像的灰度、增强图像的对比度，其应用十分广泛。

本文基于 OpenCV-Python 和 scikit-image 数字图像处理库，对图像分别进行了直方图绘制、直方图均衡（Histogram Equalization）、直方图匹配（Histogram Matching）、局部直方图增强（Local Histogram Enhancement）和直方图分割（Histogram-Based Image Segmentation）等操作。经直方图变换，部分图像的对比度有明显提升，一些图像的亮度缺陷也被显著修正。

注：由于本次实验报告页数较多，故增添目录以方便索引。

# 目　录

# 一 题目 1

## 1.1 题目

把附件图像的直方图画出。

## 1.2 解答

图像的直方图（histogram）是直方图的一种类型，作为数字图像中色调分布的图形表示。它绘制了每个色调值的像素数。通过查看特定图像的直方图，观看者将能够一目了然地判断整个色调分布。许多现代数码相机上都有图像直方图。摄影师可以使用它们作为一种辅助手段，以显示所捕获的色调分布，以及图像细节是否因高光部分被炸毁或阴影部分被涂黑而丢失。

直方图的横轴代表色调变化，而纵轴代表该特定色调的像素总数。横轴的左边代表暗部，中间代表中间色调值，右边代表亮部。纵轴表示在这些区域中的每一个区域所捕捉到的面积（像素总数）的大小。因此，一个非常暗的图像的直方图将有大部分数据点在图的左边和中间。相反，一个非常明亮的图像，很少有黑暗区域和/或阴影，其大部分数据点将在图形的右侧和中心。

以下通过 Matplotlib 中的 hist()函数，将题目中所有的原图及其直方图对应列出。

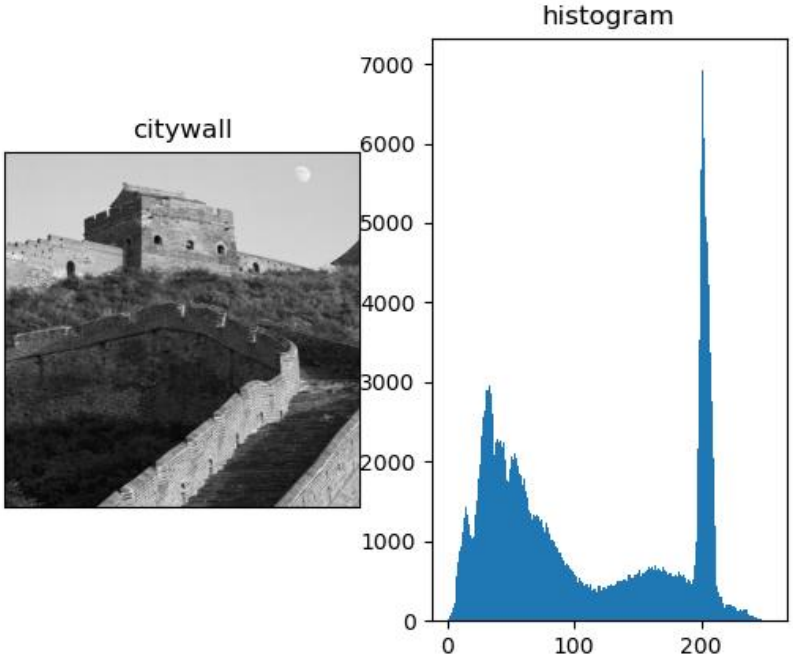


**图1　citywall 及其直方图**

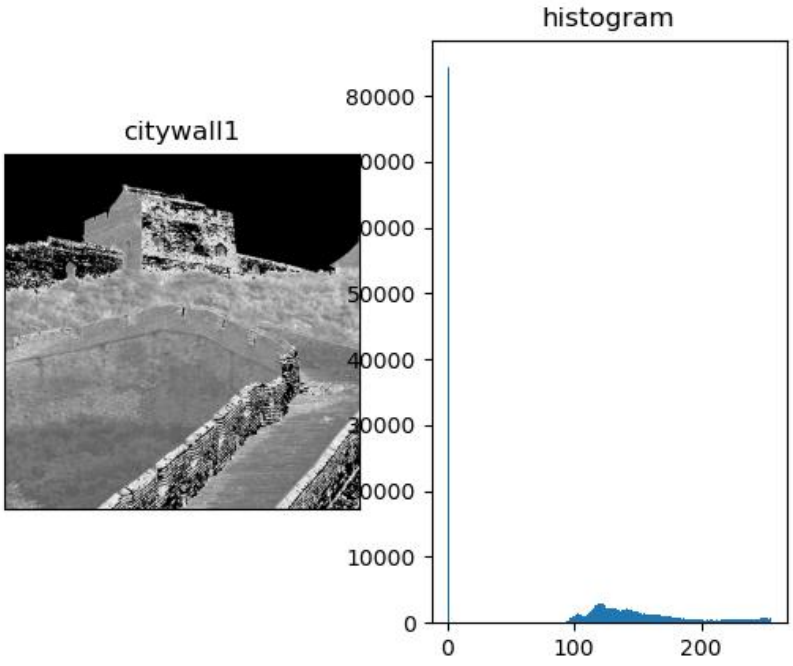


**图2　citywall1 及其直方图**

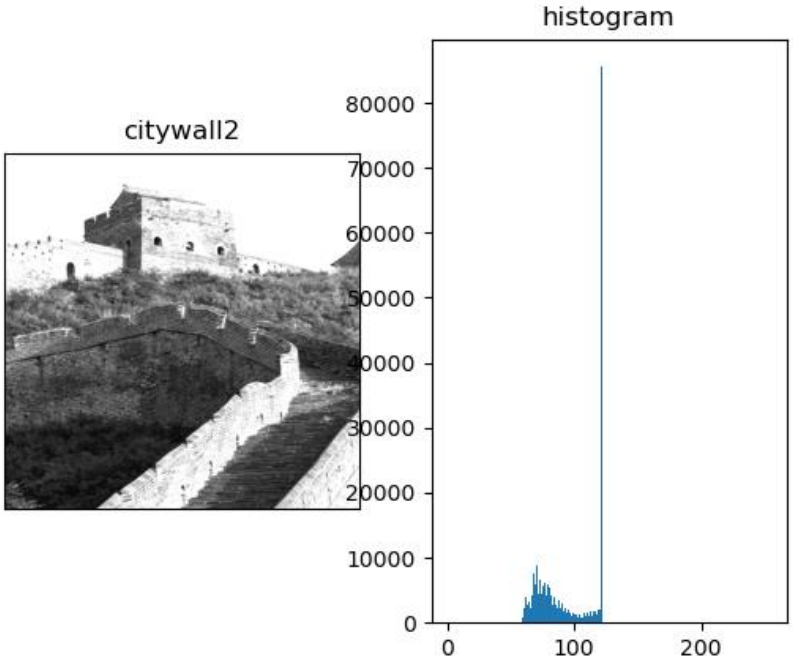


**图3　citywall2 及其直方图**

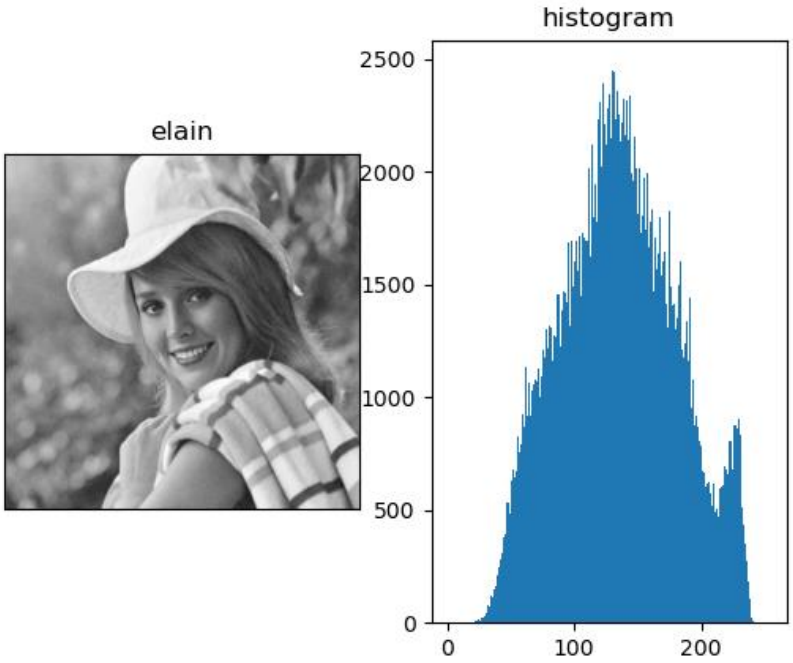


**图4　elain 及其直方图**

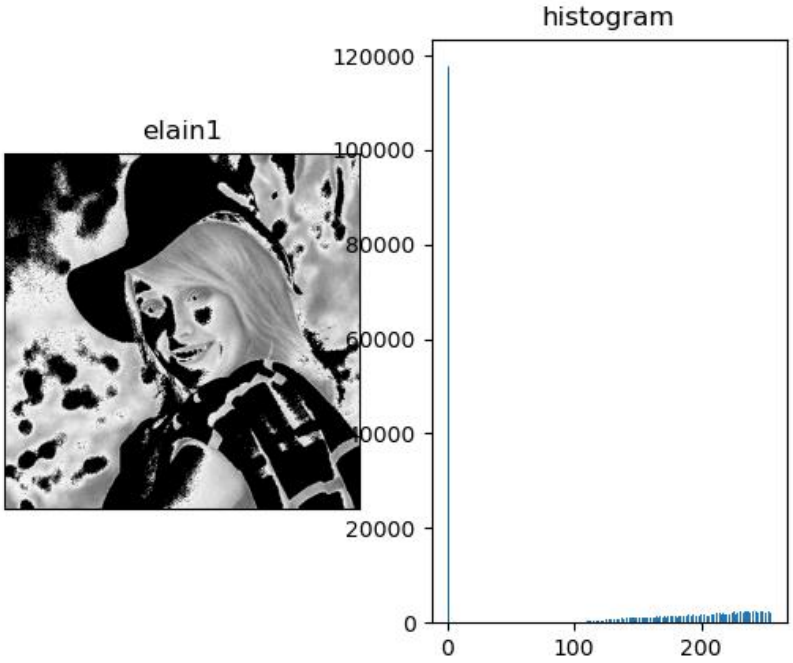


**图5　elain1 及其直方图**

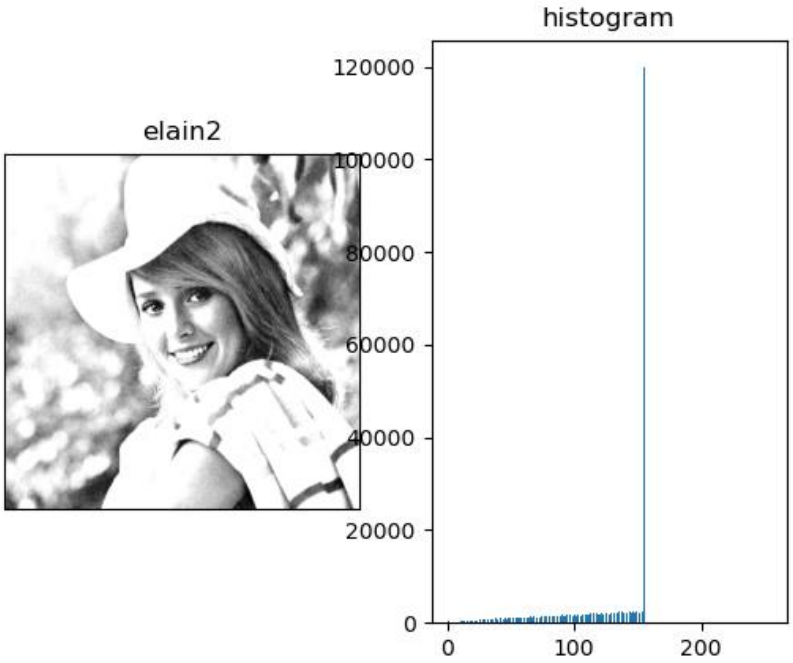


**图6　elain2 及其直方图**

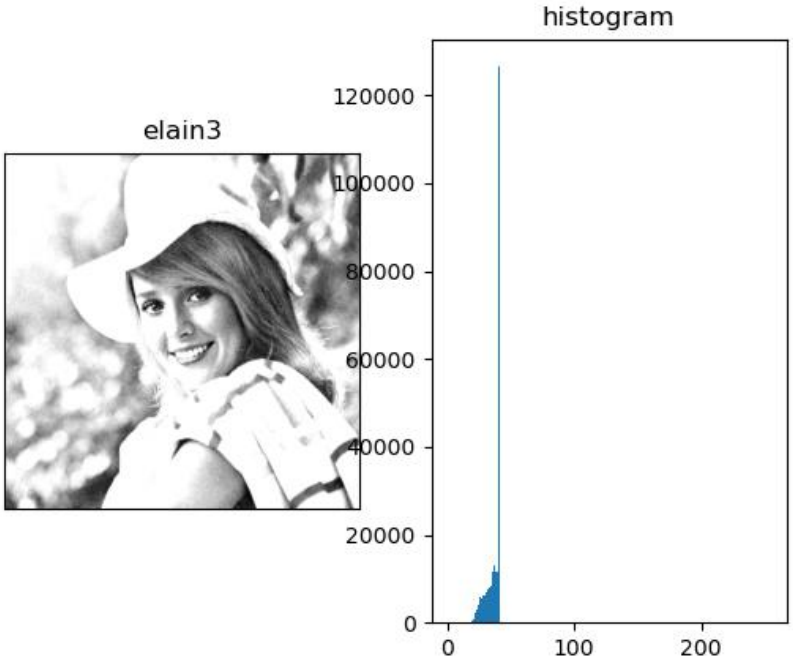


**图7　elain3 及其直方图**

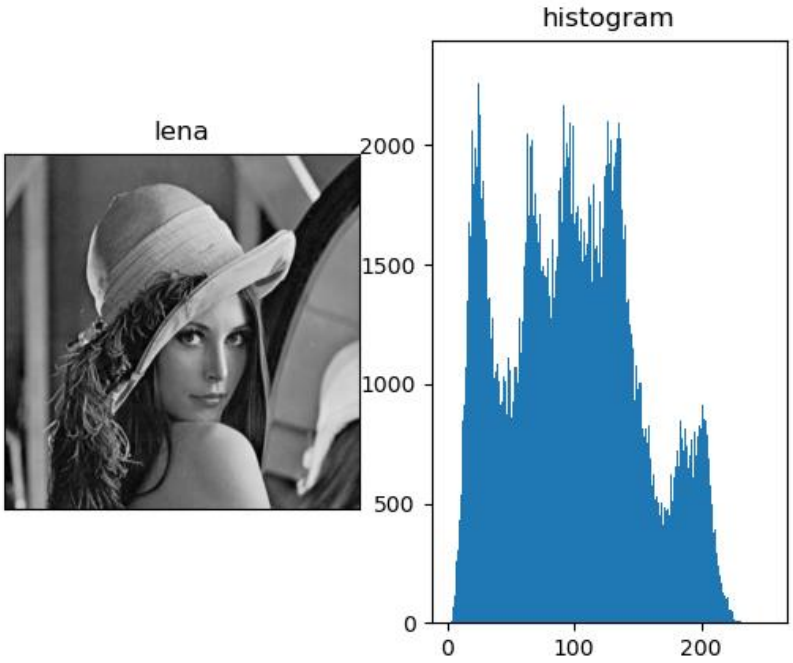


**图8　lena 及其直方图**

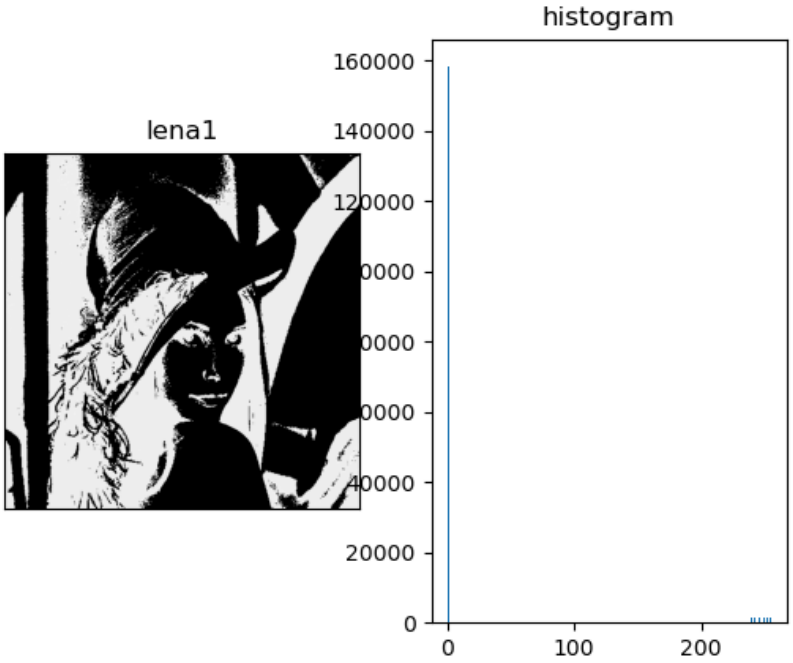


**图9　lena1 及其直方图**

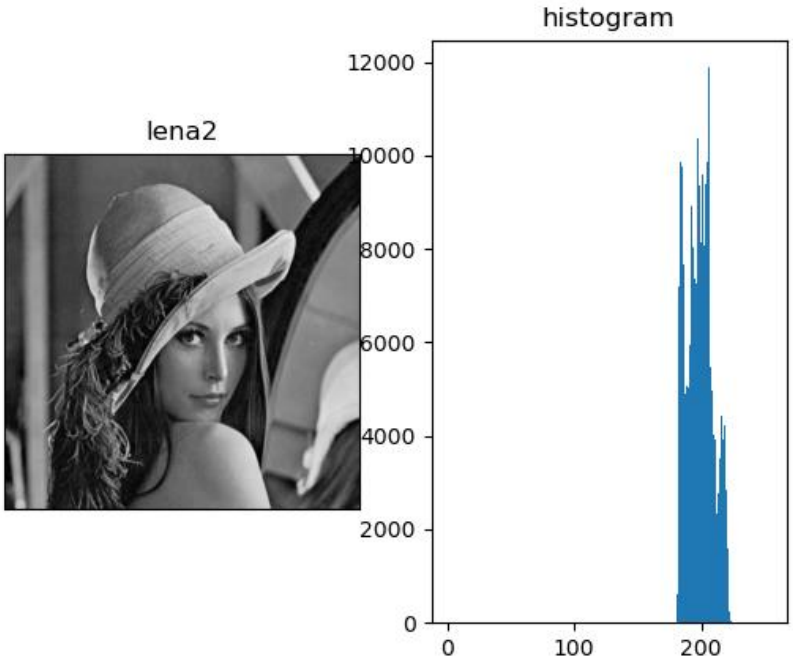


**图10　lena2 及其直方图**

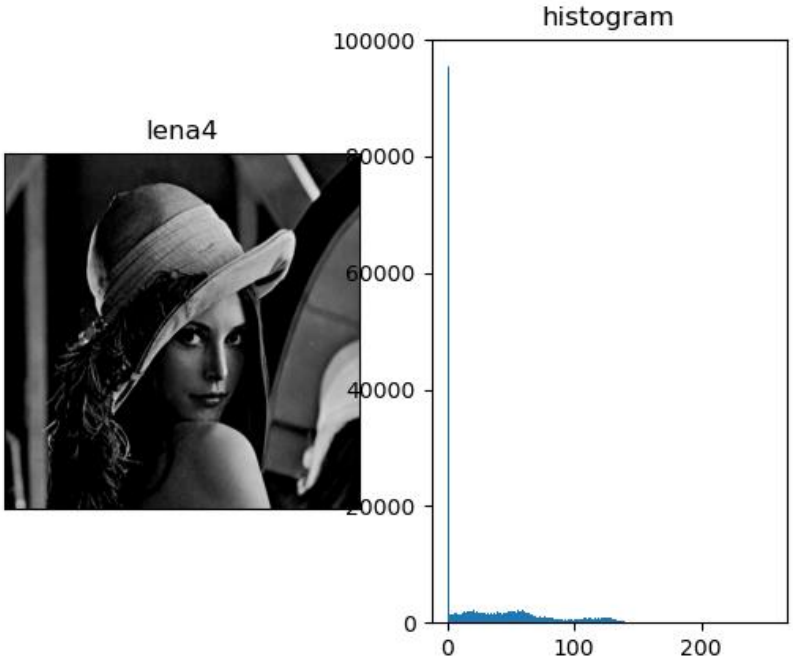


**图11　lena4 及其直方图**

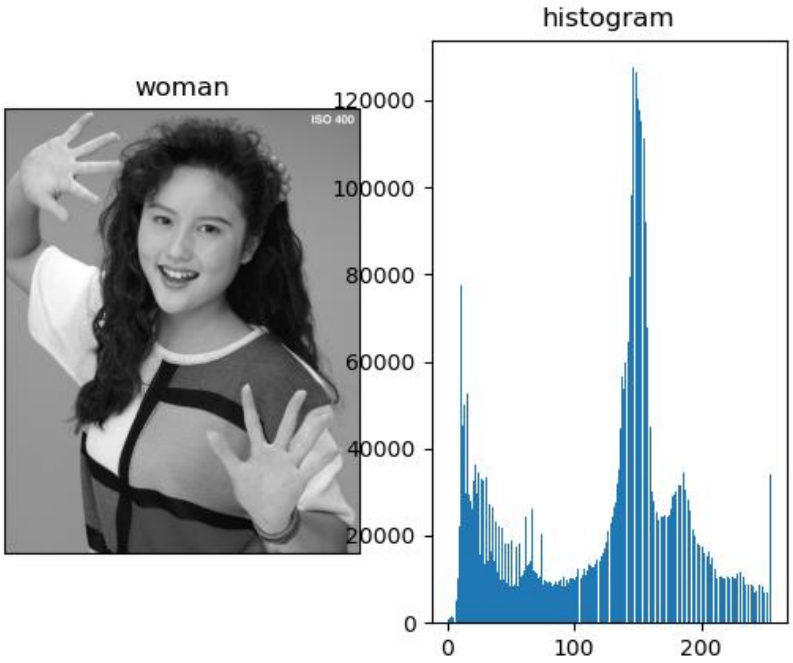


**图12　woman 及其直方图**

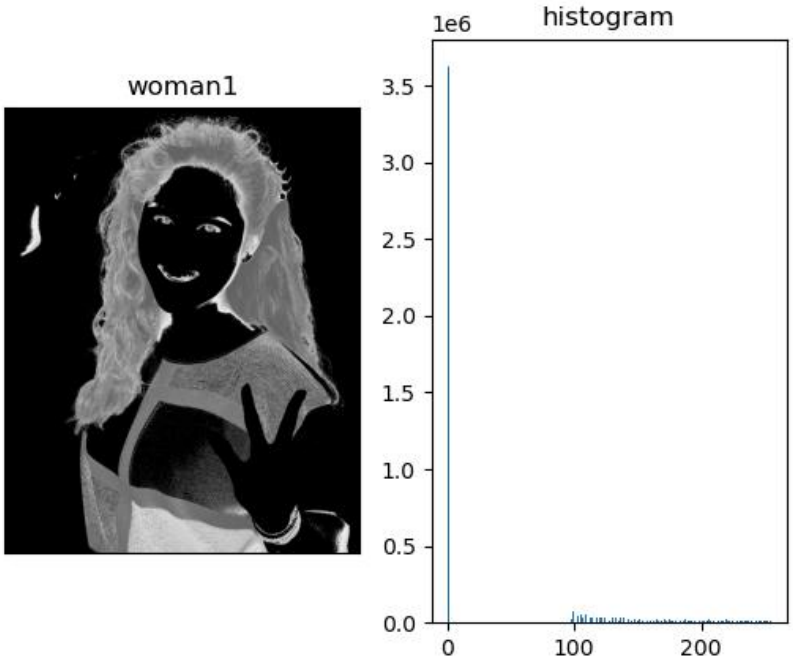


**图13　woman1 及其直方图**

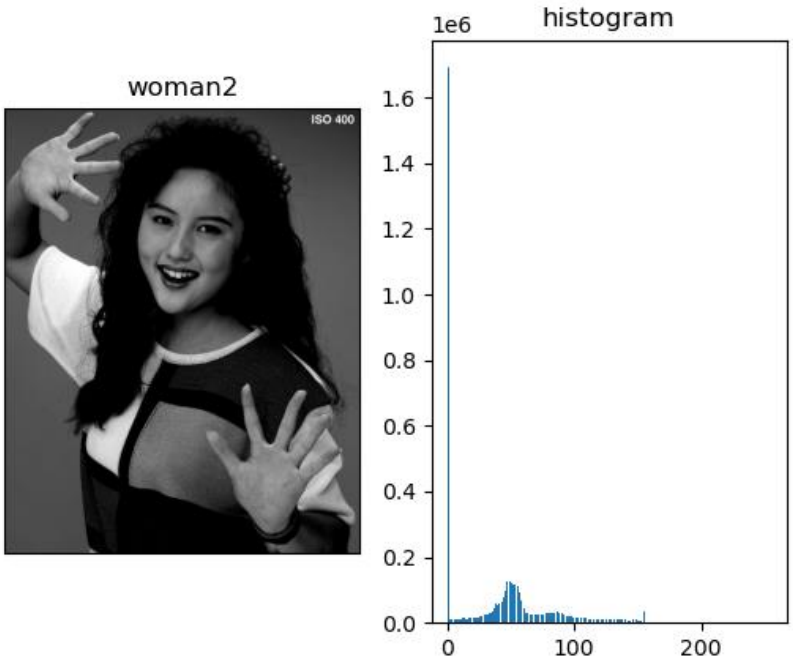


**图14　woman2 及其直方图**

# 二 题目 2

## 2.1 题目

把所有图像进行直方图均衡，输出均衡后的图像和原图像进行比对，分析改善内容。、

## 2.2 解答

直方图均衡（Image Equalization）是利用图像的直方图进行对比度调整的一种图像处理方法。

在直方图均衡中，首先计算原图像的直方图$H$，接着对直方图进行规格化，使直方图单元之和为 255，再计算直方图的积分：

$$H_i' = \sum_{0 \leqslant j < i} H(j)$$

最后使用$H'$作为查找表对图像进行变换：

$$\mathrm{dst}(x,\ y) = H'(\mathrm{src}(x,\ y))$$

在 OpenCV 中，我们使用 equalizeHist()函数对图像进行直方图均衡，以下将题目中所有的原图及其直方图均衡后的图像对应列出。

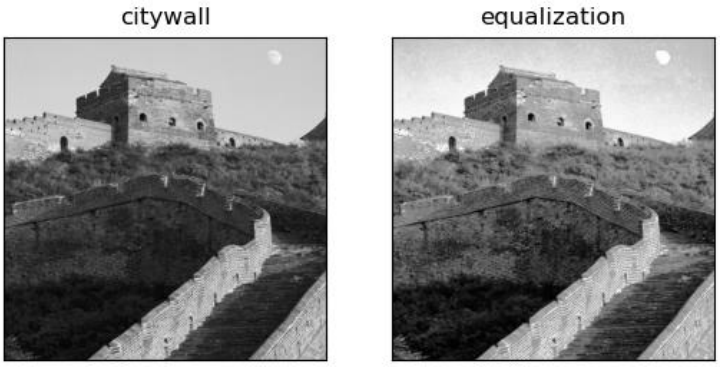


**图15　citywall 及其均衡后图像**

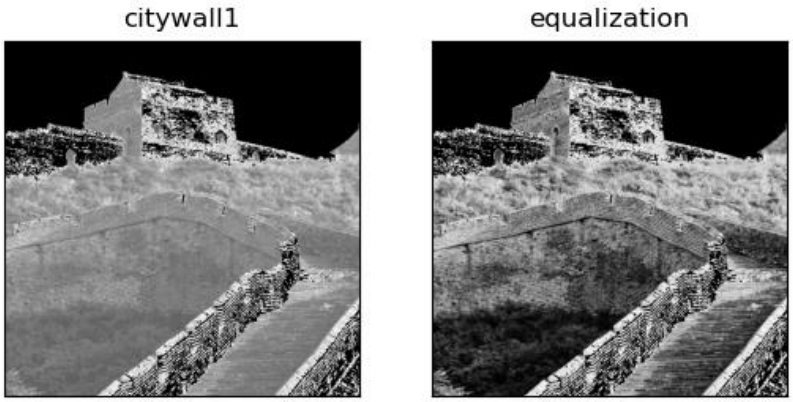


**图16　citywall1 及其均衡后图像**

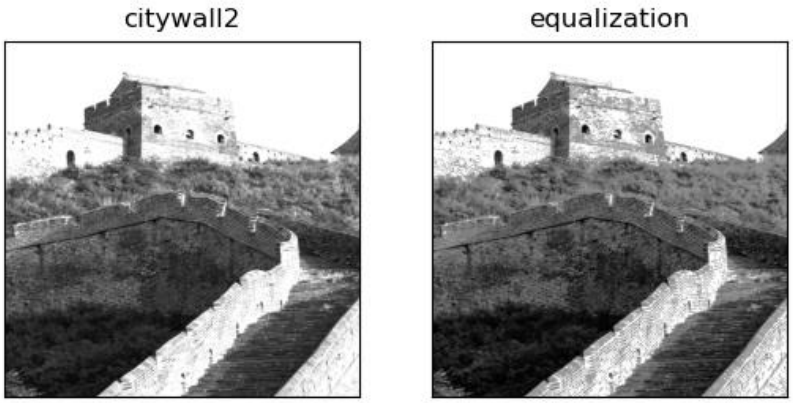


**图17　citywall2 及其均衡后图像**

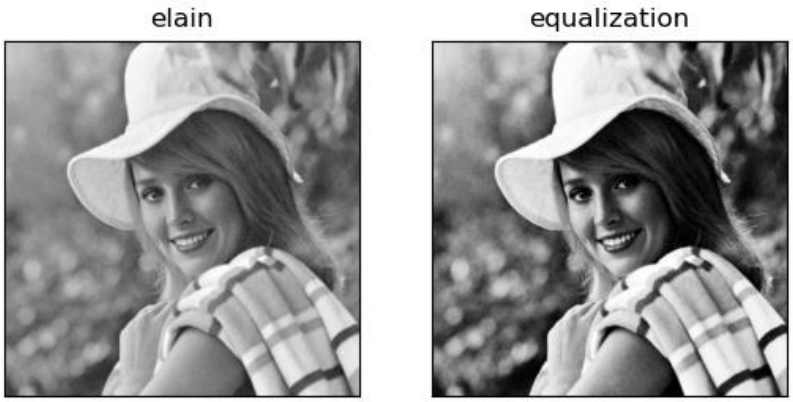


**图18　elain 及其均衡后图像**

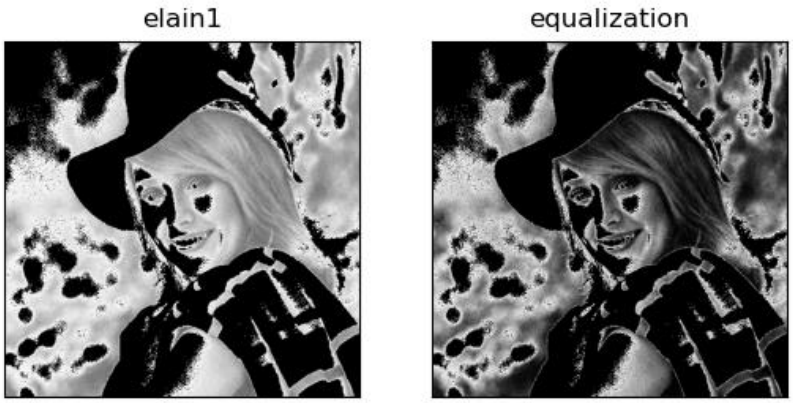


**图19　elain1 及其均衡后图像**

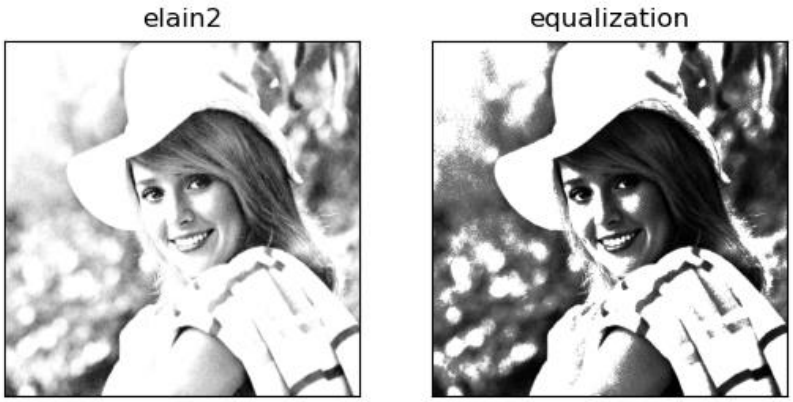


**图20　elain2 及其均衡后图像**

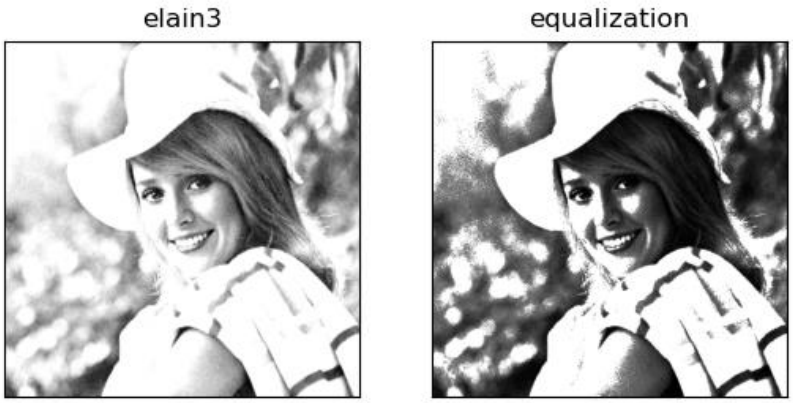


**图21　elain3 及其均衡后图像**


**图22　lena 及其均衡后图像**


**图23　lena1 及其均衡后图像**


**图24　lena2 及其均衡后图像**


**图25　lena4 及其均衡后图像**

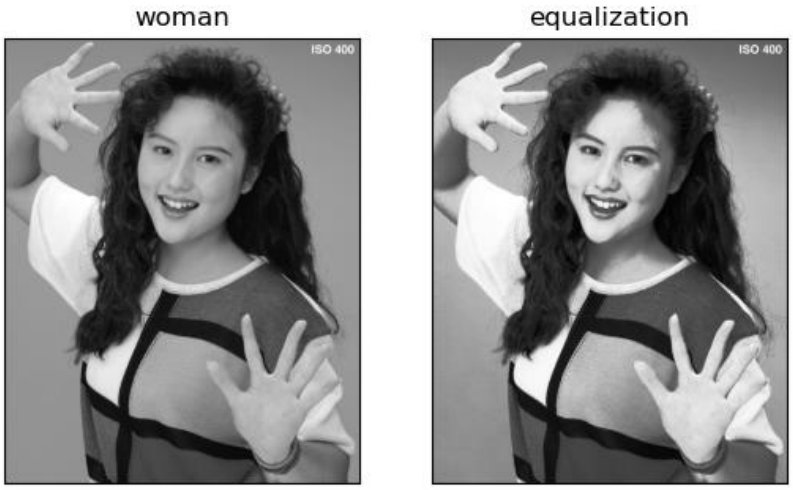


**图26　woman 及其均衡后图像**

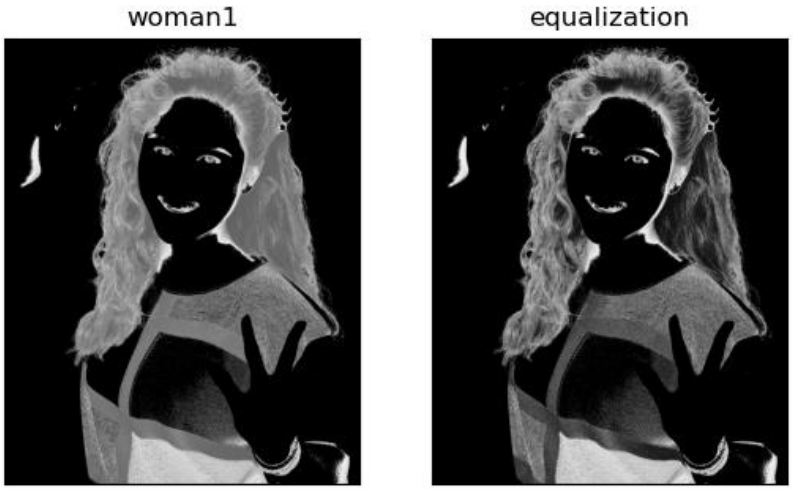


**图27　woman1 及其均衡后图像**


**图28　woman2 及其均衡后图像**

经过以上图像的对比可以表明，直方图均衡的算法可以将亮度标准化，并增加图像的对比度。事实上，这种方法通常会增加许多图像的全局对比度，特别是当图像用一个狭窄的灰度值范围来表示时。通过这种调整，可以使直方图上的亮度分布更加均匀。这样可以使局部对比度较低的区域获得较高的对比度。直方图均衡通过有效地分散用于降低图像对比度的高密集度值来实现这一点。

# 三 题目 3

## 3.1 题目

进一步把图像按照对源图像直方图的观察，各自自行指定不同源图像的直方图，进行直方图匹配，进行图像增强。

## 3.2 解答

在图像处理中，直方图匹配或直方图规范是图像的转换，使其直方图与指定的直方图匹配。直方图均衡即为其特殊情况，其中指定的直方图是均匀分布的。

例如，相对探测器校准技术可以使用直方图匹配来平衡探测器的响应。当图像是在同一位置的同一局部照明（如阴影）下，但由不同的传感器、大气条件或全局照明获取时，它可用于对两幅图像进行归一化处理。

考虑灰度输入图像$\mathbf{X}$。它具有概率密度函数$p_r(r)$，其中$r$是灰度值，而$p_r(r)$是该值的概率。这个概率可以很容易地从图像的直方图中计算出来：

$$p_r(r_j)=\frac{n_j}{n}$$

其中$n_j$是灰度值$r_j$的频率，$n$是图像中的像素总数。

现在考虑一个期望的输出概率密度函数$p_z(z)$，需要对$p_r(r)$进行转换才能将其转换为$p_z(z)$。每个概率密度函数（PDF）都可以映射到它的累积分布函数（CDF）：

$$S(r_k)=\sum_{j=0}^{k}p_r(r_j),\qquad k=0,1,2,3,...,L-1$$

$$G(z_k)=\sum_{j=0}^{k}p_z(z_j),\qquad k=0,1,2,3,...,L-1$$

其中$L$是灰度级的总数（标准图像为 256）。其思想是将$\mathbf{X}$中的每个$r$值映射到所需概率密度函数中具有相同概率的$z$值，即$S(r_j)=G(z_i)\ \Rightarrow z=G^{-1}(S(r))$。

以下使用 citywall.bmp 为源图像，分别使用 citywall1.bmp、citywall2.bmp 对其进行直方图匹配。

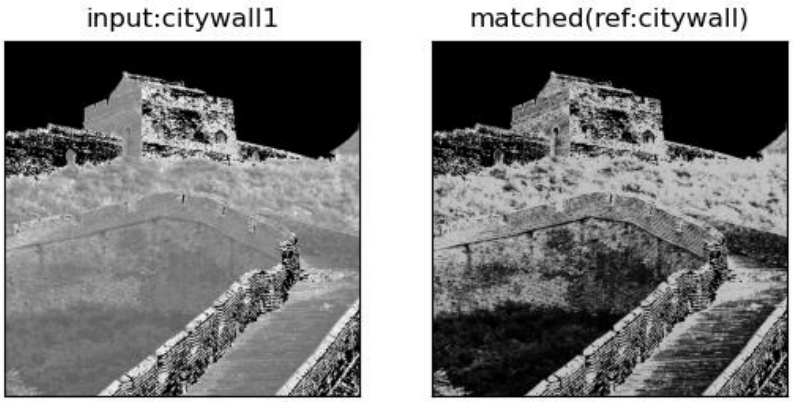


**图29　citywall1 直方图匹配结果（参考图像为 citywall）**

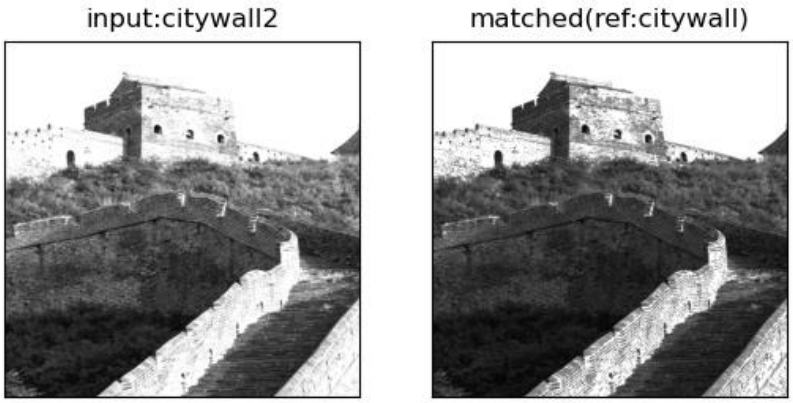


**图30　citywall2 直方图匹配结果（参考图像为 citywall）**

以下使用 elain.bmp 为源图像，分别使用 elain1.bmp、elain2.bmp 和 elain3.bmp 对其进行直方图匹配。

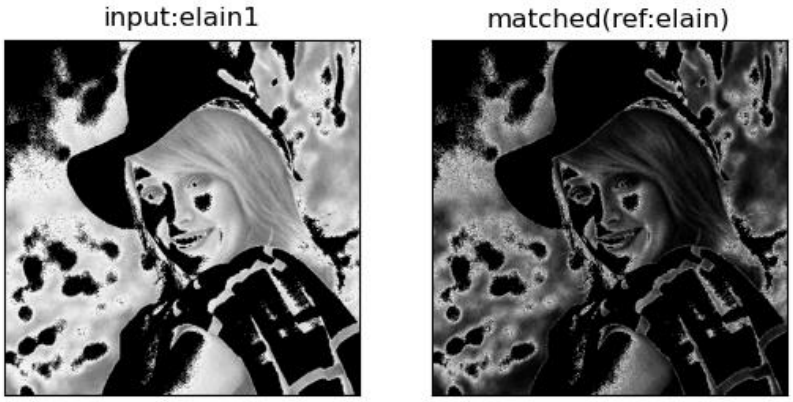


**图31　elain1 直方图匹配结果（参考图像为 elain）**

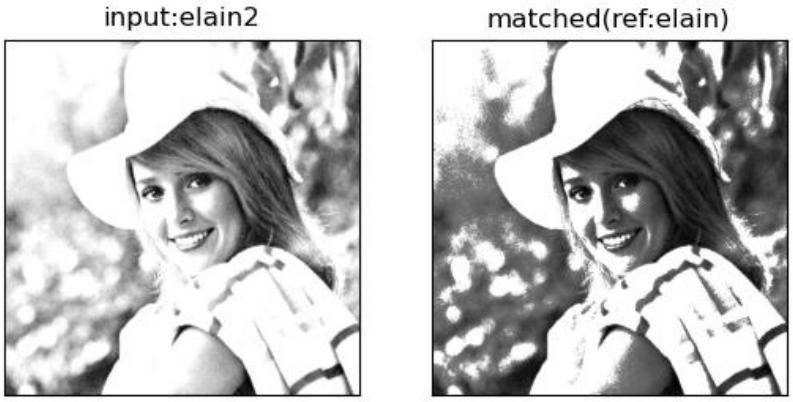


**图32　elain2 直方图匹配结果（参考图像为 elain）**

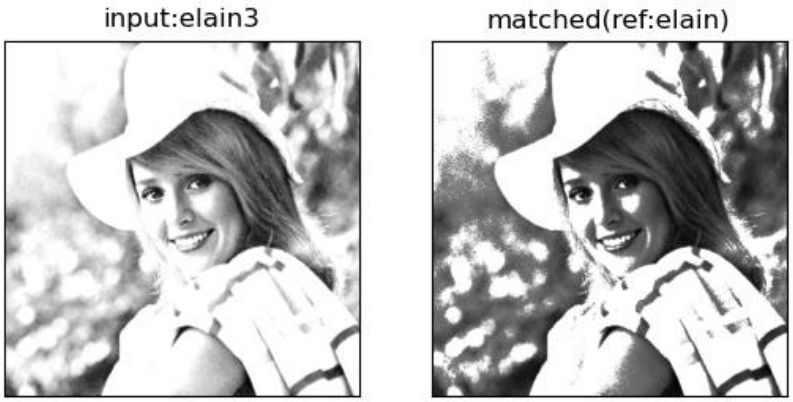


**图33　elain3 直方图匹配结果（参考图像为 elain）**

以下使用 lena.bmp 为源图像，分别使用 lena1.bmp、lena2.bmp 和 lena4.bmp 对其进行直方图匹配。

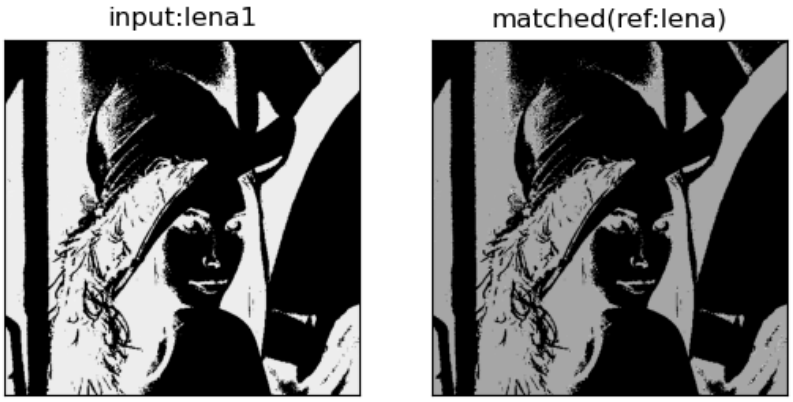


**图34　lena1 直方图匹配结果（参考图像为 lena）**


**图35　lena2 直方图匹配结果（参考图像为 lena）**


**图36　lena4 直方图匹配结果（参考图像为 lena）**

以下使用 woman.bmp 为源图像，分别使用 woman1.bmp、woman2.bmp 对其进行直方图匹配。

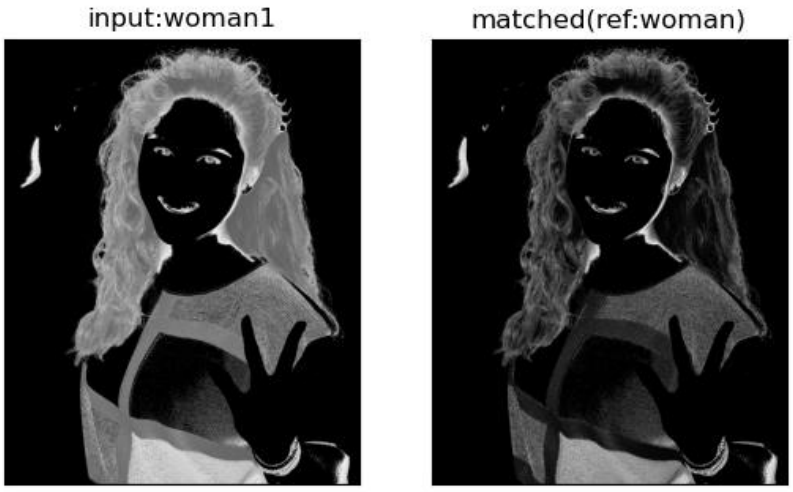


**图37　woman1 直方图匹配结果（参考图像为 woman）**


**图38　woman2 直方图匹配结果（参考图像为 woman）**

可见，经过直方图匹配后，图像的对比度略有提升，但是仍然有很多地方的细节模糊不清，除此以外，有些部分过分增强，使得难以辨识。

# 四 题目 4

## 4.1 题目

对 elain 和 lena 图像进行7×7的局部直方图增强。

## 4.2 解答

在传统的直方图均衡后，某些图片的背景对比度确实有所改善。但如果一些图片亮度过高，可能会丢失哪些部分的大部分信息。

为了解决这个问题，我们采用局部直方图均衡。在此情形下，图像被划分成为小块。然后像往常一样对每个块进行直方图均衡。因此，在一个小区域内，直方图将局限于一个小区域（除非有噪声）。如果有噪音，它会被放大。为了避免这种情况，我们进一步采用自适应直方图增强均衡技术（Contrast Limited Adaptive Histogram Equalization），在此处应用对比度限制。如果任何直方图单元高于指定的对比度限制（在 OpenCV 中默认为40），则在应用直方图均衡化之前，这些像素将被剪裁并均匀分布到其他单元。均衡后，为了消除平铺边界中的瑕疵，将应用双线性插值。

如下为使用局部直方图增强后的结果，可见其结果优于传统的直方图均衡。

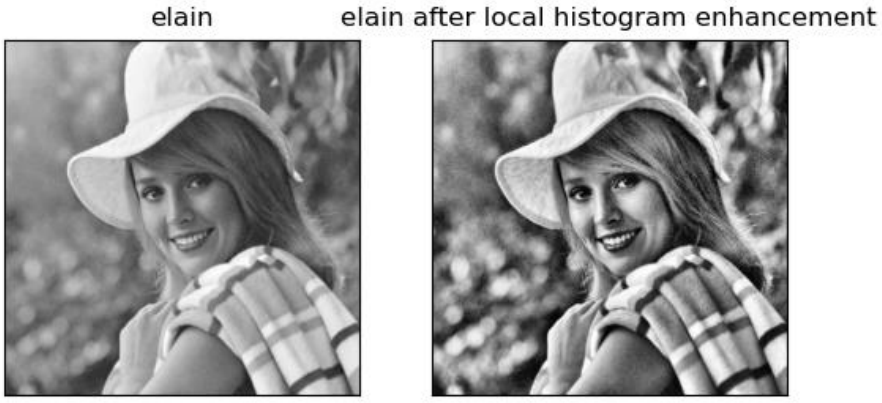


**图39　elain 局部直方图增强**


**图40　lena 局部直方图增强**

# 五 题目 5

## 5.1 题目

利用直方图对图像 elain 和 woman 进行分割。

## 5.2 解答

在 scikit-image 中，为了对比各算法优劣，我们先采用不同的算法对图像进行二值切割，其切割结果如下图所示。


**图41　使用不同图像切割算法对 elain 进行切割**

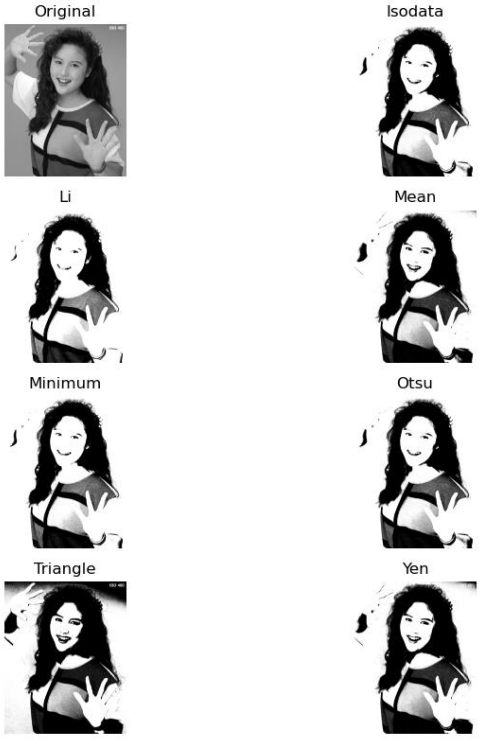


**图42　使用不同图像切割算法对 woman 进行切割**

经观察可发现，在所有***基于直方图***的图像分割算法中，使用大津算法（Otsu's method）的图像分割效果较好，因此使用该算法对图像进行直方图分割。

该算法穷尽搜索使类内方差最小化的阈值，定义为两类方差的加权和：

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t)$$

权重$\omega_0$和$\omega_1$是两个类被阈值$t$分隔的概率，$\sigma_0{}^2$和$\sigma_1{}^2$是这两个类的方差。类概率$\omega_{0,1}(t)$是从直方图的$L$个单元计算出来的：

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i),\ \omega_1(t) = \sum_{i=t}^{L-1} p(i)$$

对于两个类而言，最小化类内方差等价于最大化类间方差：

$$\sigma_b^2(t) = \sigma^2 - \sigma_w^2(t) = \omega_0(t)(\mu_0 - \mu_T)^2 + \omega_1(t)(\mu_1 - \mu_T)^2 = \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2$$

其中$\omega$为类别概率，$\mu$为类均值，且

$$\mu_0(t)=\frac{\sum_{i=0}^{t-1} i\cdot p(i)}{\omega_0(t)},\ \mu_1(t)=\frac{\sum_{i=t}^{L-1} i\cdot p(i)}{\omega_1(t)},\ \mu_T=\sum_{i=0}^{L-1} i\cdot p(i)$$

类概率和类均值可以迭代计算，于是产生如下算法：

计算直方图和每个灰度级别的概率；

设置初始的$\omega_i(0)$和$\mu_i(0)$；

逐步迭代所有可能的阈值$t=1,...$最大强度：更新$\omega_i$和$\mu_i$，并计算${\sigma_b}^2(t)$；

所需的阈值对应于${\sigma_b}^2(t)$的最大值。

如下为使用 Otsu 算法分别对 elain 和 woman 图像进行直方图切割的结果。

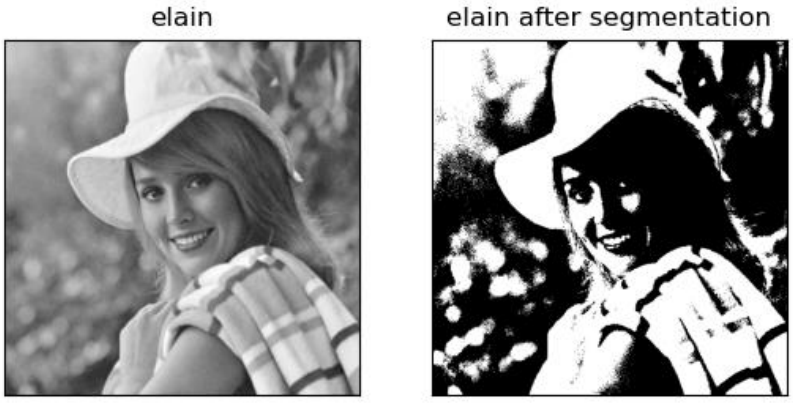


**图43　使用 Otsu 算法分割 elain**


**图44　使用 Otsu 算法分割 woman**

# 六 附录：所有代码

## 6.1 导入所需库文件

```
import cv2

from skimage.filters import try_all_threshold
from skimage.filters import threshold_otsu
from skimage.exposure import match_histograms
import matplotlib.pyplot as plt
```

## 6.2 读取图片文件

```
# %% ================================================读取所有图形文

==========================================================

citywall = cv2.cvtColor(cv2.imread("Resource/citywall.bmp",
flags=cv2.IMREAD_COLOR), code=cv2.COLOR_BGR2GRAY)
citywall1 = cv2.cvtColor(cv2.imread("Resource/citywall1.bmp",
flags=cv2.IMREAD_COLOR), code=cv2.COLOR_BGR2GRAY)
citywall2 = cv2.cvtColor(cv2.imread("Resource/citywall2.bmp",
flags=cv2.IMREAD_COLOR), code=cv2.COLOR_BGR2GRAY)

elain = cv2.cvtColor(cv2.imread("Resource/elain.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
elain1 = cv2.cvtColor(cv2.imread("Resource/elain1.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
elain2 = cv2.cvtColor(cv2.imread("Resource/elain2.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
elain3 = cv2.cvtColor(cv2.imread("Resource/elain3.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)

lena = cv2.cvtColor(cv2.imread("Resource/lena.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
lena1 = cv2.cvtColor(cv2.imread("Resource/lena1.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
lena2 = cv2.cvtColor(cv2.imread("Resource/lena2.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
lena4 = cv2.cvtColor(cv2.imread("Resource/lena4.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
```

```
woman = cv2.cvtColor(cv2.imread("Resource/woman.BMP", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
woman1 = cv2.cvtColor(cv2.imread("Resource/woman1.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
woman2 = cv2.cvtColor(cv2.imread("Resource/woman2.bmp", flags=cv2.IMREAD_COLOR),
code=cv2.COLOR_BGR2GRAY)
```

## 6.3 题目 1：绘制直方图

```
# ====================================================绘制直方

============================================================
# %% citywall
plt.subplot(121), plt.imshow(citywall, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("citywall")
plt.subplot(122), plt.hist(citywall.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% citywall1
plt.subplot(121), plt.imshow(citywall1, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("citywall1")
plt.subplot(122), plt.hist(citywall1.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% citywall2
plt.subplot(121), plt.imshow(citywall2, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("citywall2")
plt.subplot(122), plt.hist(citywall2.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% elain
plt.subplot(121), plt.imshow(elain, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("elain")
plt.subplot(122), plt.hist(elain.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% elain1
```

```
plt.subplot(121), plt.imshow(elain1, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("elain1")
plt.subplot(122), plt.hist(elain1.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% elain2
plt.subplot(121), plt.imshow(elain2, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("elain2")
plt.subplot(122), plt.hist(elain2.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% elain3
plt.subplot(121), plt.imshow(elain3, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("elain3")
plt.subplot(122), plt.hist(elain3.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% lena
plt.subplot(121), plt.imshow(lena, cmap='gray'), plt.xticks([]), plt.yticks([]),
plt.title("lena")
plt.subplot(122), plt.hist(lena.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% lena1
plt.subplot(121), plt.imshow(lena1, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("lena1")
plt.subplot(122), plt.hist(lena1.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% lena2
plt.subplot(121), plt.imshow(lena2, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("lena2")
plt.subplot(122), plt.hist(lena2.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% lena4
```

```
plt.subplot(121), plt.imshow(lena4, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("lena4")
plt.subplot(122), plt.hist(lena4.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% woman
plt.subplot(121), plt.imshow(woman, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("woman")
plt.subplot(122), plt.hist(woman.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% woman1
plt.subplot(121), plt.imshow(woman1, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("woman1")
plt.subplot(122), plt.hist(woman1.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()

# %% woman2
plt.subplot(121), plt.imshow(woman2, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("woman2")
plt.subplot(122), plt.hist(woman2.ravel(), bins=256, range=(0, 255)),
plt.title("histogram")
plt.show()
```

## 6.4 题目 2：直方图均衡

```
# ================================histogram equalization(直方图
衡)=====================================================
# %% citywall
plt.subplot(121), plt.imshow(citywall, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("citywall")
plt.subplot(122), plt.imshow(cv2.equalizeHist(citywall), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()
```

```
# %% citywall1
plt.subplot(121), plt.imshow(citywall1, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("citywall1")
plt.subplot(122), plt.imshow(cv2.equalizeHist(citywall1), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% citywall2
plt.subplot(121), plt.imshow(citywall2, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("citywall2")
plt.subplot(122), plt.imshow(cv2.equalizeHist(citywall2), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% elain
plt.subplot(121), plt.imshow(elain, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("elain")
plt.subplot(122), plt.imshow(cv2.equalizeHist(elain), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% elain1
plt.subplot(121), plt.imshow(elain1, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("elain1")
plt.subplot(122), plt.imshow(cv2.equalizeHist(elain1), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% elain2
plt.subplot(121), plt.imshow(elain2, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("elain2")
plt.subplot(122), plt.imshow(cv2.equalizeHist(elain2), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% elain3
plt.subplot(121), plt.imshow(elain3, cmap='gray'), plt.xticks([]),
```

```
plt.yticks([]), plt.title("elain3")
plt.subplot(122), plt.imshow(cv2.equalizeHist(elain3), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% lena
plt.subplot(121), plt.imshow(lena, cmap='gray'), plt.xticks([]), plt.yticks([]),
plt.title("lena")
plt.subplot(122), plt.imshow(cv2.equalizeHist(lena), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% lena1
plt.subplot(121), plt.imshow(lena1, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("lena1")
plt.subplot(122), plt.imshow(cv2.equalizeHist(lena1), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% lena2
plt.subplot(121), plt.imshow(lena2, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("lena2")
plt.subplot(122), plt.imshow(cv2.equalizeHist(lena2), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% lena4
plt.subplot(121), plt.imshow(lena4, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("lena4")
plt.subplot(122), plt.imshow(cv2.equalizeHist(lena4), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% woman
plt.subplot(121), plt.imshow(woman, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("woman")
plt.subplot(122), plt.imshow(cv2.equalizeHist(woman), cmap='gray'),
```

```
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% woman1
plt.subplot(121), plt.imshow(woman1, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("woman1")
plt.subplot(122), plt.imshow(cv2.equalizeHist(woman1), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()

# %% woman2
plt.subplot(121), plt.imshow(woman2, cmap='gray'), plt.xticks([]),
plt.yticks([]), plt.title("woman2")
plt.subplot(122), plt.imshow(cv2.equalizeHist(woman2), cmap='gray'),
plt.xticks([]), plt.yticks([]), plt.title(
    "equalization")
plt.show()
```

## 6.5 题目 3：直方图匹配

```
# ==========================================histogram matching(直方图
配)==========================================
# %% 图像显示函数
def display_image(images, titles):
    for i in range(2):
        plt.subplot(1, 2, i + 1)
        plt.imshow(images[i], 'gray')
        plt.title(titles[i])
        plt.xticks([]), plt.yticks([])
    plt.show()


# %% citywall1 匹配 citywall
matchCitywall1WithCitywall = match_histograms(image=citywall1,
reference=citywall, multichannel=False)
images = [citywall1, matchCitywall1WithCitywall]
titles = ['input:citywall1', 'matched(ref:citywall)']
```

```
display_image(images, titles)

# %% citywall2 匹配 citywall
matchCitywall2WithCitywall = match_histograms(image=citywall2,
reference=citywall, multichannel=False)
images = [citywall2, matchCitywall2WithCitywall]
titles = ['input:citywall2', 'matched(ref:citywall)']
display_image(images, titles)

# %% elain1 匹配 elain
matchElain1WithElain = match_histograms(image=elain1, reference=elain,
multichannel=False)
images = [elain1, matchElain1WithElain]
titles = ['input:elain1', 'matched(ref:elain)']
display_image(images, titles)

# %% elain2 匹配 elain
matchElain2WithElain = match_histograms(image=elain2, reference=elain,
multichannel=False)
images = [elain2, matchElain2WithElain]
titles = ['input:elain2', 'matched(ref:elain)']
display_image(images, titles)

# %% elain3 匹配 elain
matchElain3WithElain = match_histograms(image=elain3, reference=elain,
multichannel=False)
images = [elain3, matchElain3WithElain]
titles = ['input:elain3', 'matched(ref:elain)']
display_image(images, titles)

# %% lena1 匹配 lena
matchLena1WithLena = match_histograms(image=lena1, reference=lena,
multichannel=False)
images = [lena1, matchLena1WithLena]
titles = ['input:lena1', 'matched(ref:lena)']
display_image(images, titles)

# %% lena2 匹配 lena
matchLena2WithLena = match_histograms(image=lena2, reference=lena,
multichannel=False)
images = [lena2, matchLena2WithLena]
titles = ['input:lena2', 'matched(ref:lena)']
```

```
display_image(images, titles)

# %% lena4 匹配 lena
matchLena4WithLena = match_histograms(image=lena4, reference=lena,
multichannel=False)
images = [lena4, matchLena4WithLena]
titles = ['input:lena4', 'matched(ref:lena)']
display_image(images, titles)

# %% woman1 匹配 woman
matchWoman1WithWoman = match_histograms(image=woman1, reference=woman,
multichannel=False)
images = [woman1, matchWoman1WithWoman]
titles = ['input:woman1', 'matched(ref:woman)']
display_image(images, titles)

# %% woman2 匹配 woman
matchWoman2WithWoman = match_histograms(image=woman2, reference=woman,
multichannel=False)
images = [woman2, matchWoman2WithWoman]
titles = ['input:woman2', 'matched(ref:woman)']
display_image(images, titles)
```

## 6.6 题目 4：局部直方图增强

```
# ====================================================局部直方图增

============================================================
# %% 局部直方图增强 elain
clahe = cv2.createCLAHE(clipLimit=3.0, tileGridSize=(7, 7))
claneElain = clahe.apply(elain)
images = [elain, claneElain]
titles = ['elain', 'elain after local histogram enhancement']
display_image(images, titles)

# %% 局部直方图增强 lena
claneLena = clahe.apply(lena)
images = [lena, claneLena]
titles = ['lena', 'lena after local histogram enhancement']
display_image(images, titles)
```

## 6.7 题目 5：直方图分割

```
# ========================================================直方图分
========================================================
# %% elain 分割（尝试）
segElain, ax = try_all_threshold(elain, figsize=(10, 8), verbose=True)
plt.show()

# %% woman 分割（尝试）
segWoman, ax = try_all_threshold(woman, figsize=(10, 8), verbose=True)
plt.show()

# %% elain 分割
thresholdElain = threshold_otsu(elain)
segElainBinary = elain > thresholdElain
images = [elain, segElainBinary]
titles = ['elain', 'elain after segmentation']
display_image(images, titles)
# %% woman 分割
thresholdWoman = threshold_otsu(woman)
segWomanBinary = woman > thresholdWoman
images = [woman, segWomanBinary]
titles = ['woman', 'woman after segmentation']
display_image(images, titles)
```