

数字图像处理

第三次作业



摘要

本报告通过了解图像直方图来增强图像的显示效果，用得到的方法有直方图均衡，直方图匹配和图像分割。直方图均衡是调整图像的对比度使其增强；直方图匹配是用目标直方图来处理原来的图像；图像分割是将一幅图像的前景和背景区分开来。

杨致远

电信钱51

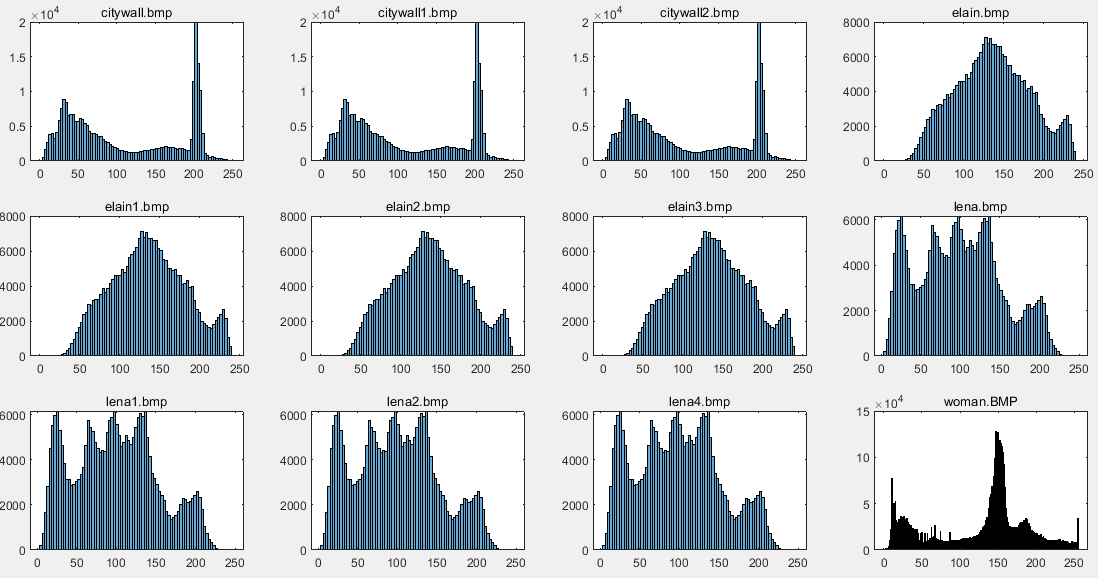
2150405061

2018/4/5

## 任务1—把附件图像的直方图画出

**实验介绍：**

本实验看上去是简单地画出直方图就行了，但是通过加载图片才发现每一个类型加载出来的直方图是一样的，在群里同学提醒后，我才发现这些图片的数据矩阵都是一样，但调色盘矩阵是不一样的，所以如果不加载调色盘，那么出来的直方图就是一样的，如下图所示。



【索引图片】，根据百度百科的解释，是一种把像素值直接作为RGB调色板下标的图像。索引图像可把像素值“直接映射”为调色板数值。

在这次的作业任务中，我们举个例子来说就很明白了，图像数据矩阵的（1，1）单元的内容为196，也就是说这一像素的颜色就是调色盘的196行所定义的颜色，可以看到调色盘矩阵的第196行为【0.60536，0.60536，0.60536】，表示RGB三个值的分量，且这三个分量的地位都相同。

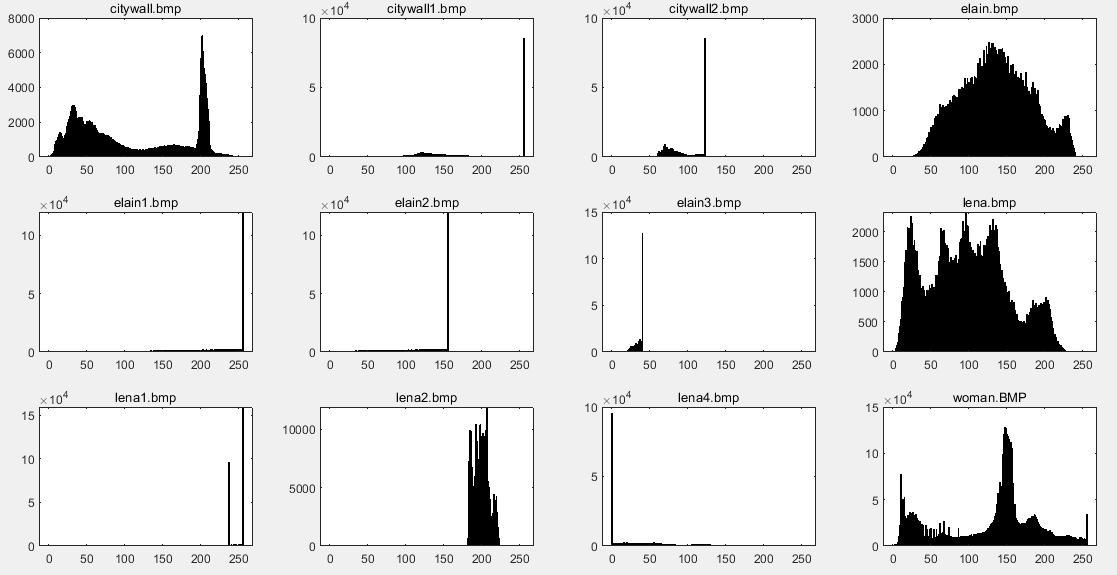
还有一个问题就是有几张图的调色盘值不是完整的256行，但这没有关系，在处理的时候，这些值默认地变成了0。使用的函数就是rgb2gray（map）就可以完成以上的加载调色盘的工作了。

在本次实验中我使用几个小技巧来节省代码空间和优化输入结果：

1. 用【file\_structure = dir('\*.bmp')】来模糊加载文件下的文件，然后就能使用for循环来批量化处理图片
2. 使用【edges】来调整直方图的边界，因为默认会有一个自适应，所以可以调整到（0~255）

我自己编写的直方图程序，可以用tabulate来统计每个灰度值出现的次数，横坐标为0~255个单点，纵坐标为统计的次数，进而画出图像。

**结果展示：**



**结果讨论：**

由上图可以看出，统一名称的图片，在不同的调色盘下，直方图的差别很大，由前面的原理分析，可以很容易地得出其中的道理。

## 任务2—直方图均衡

**实验介绍：**

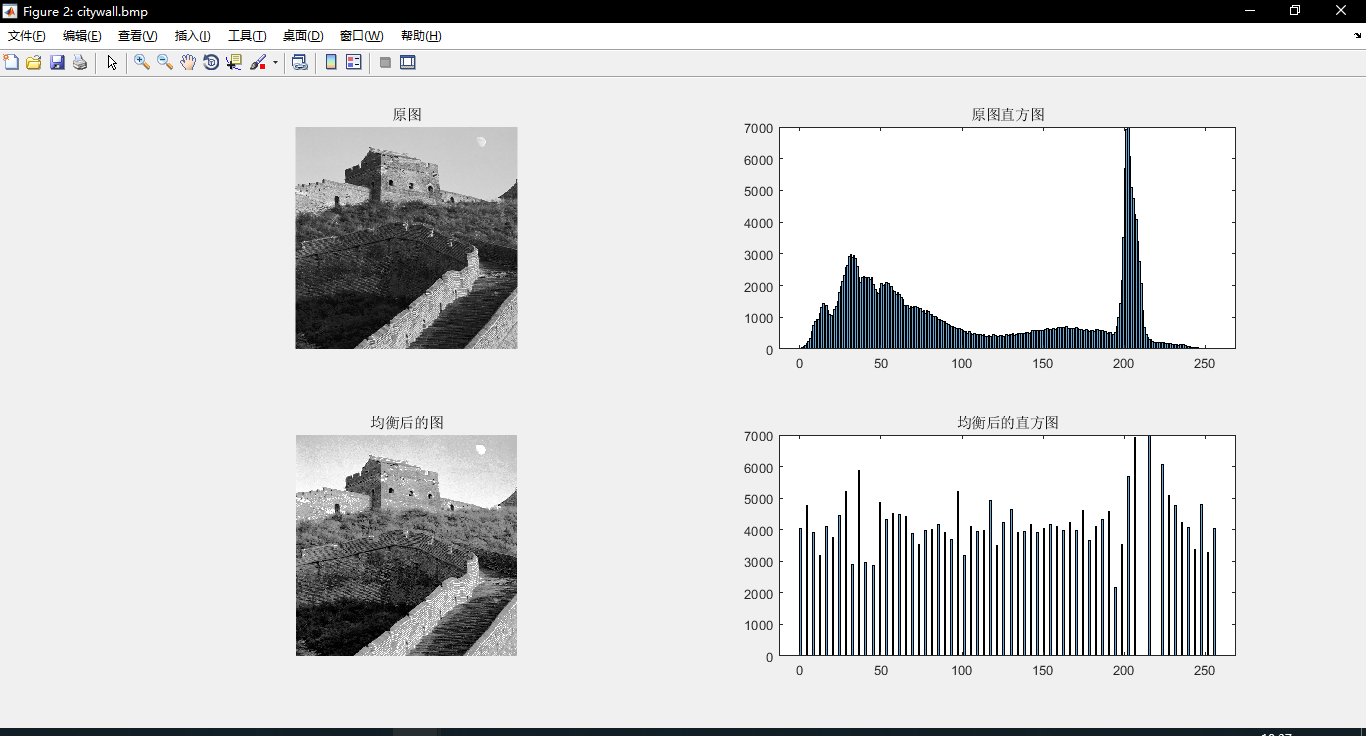
直方图均衡处理是以累计分布函数为基础的直方图修改法，就是把一已知灰度概率分布的图像经过一种变换，使之演变成一幅具有均匀灰度概率分布的新图像。直方图均衡的实现原理是先要统计每一个灰度值出现的次数，要从小到大把次数累积起来，再利用公式进行计算变换后的灰度值。

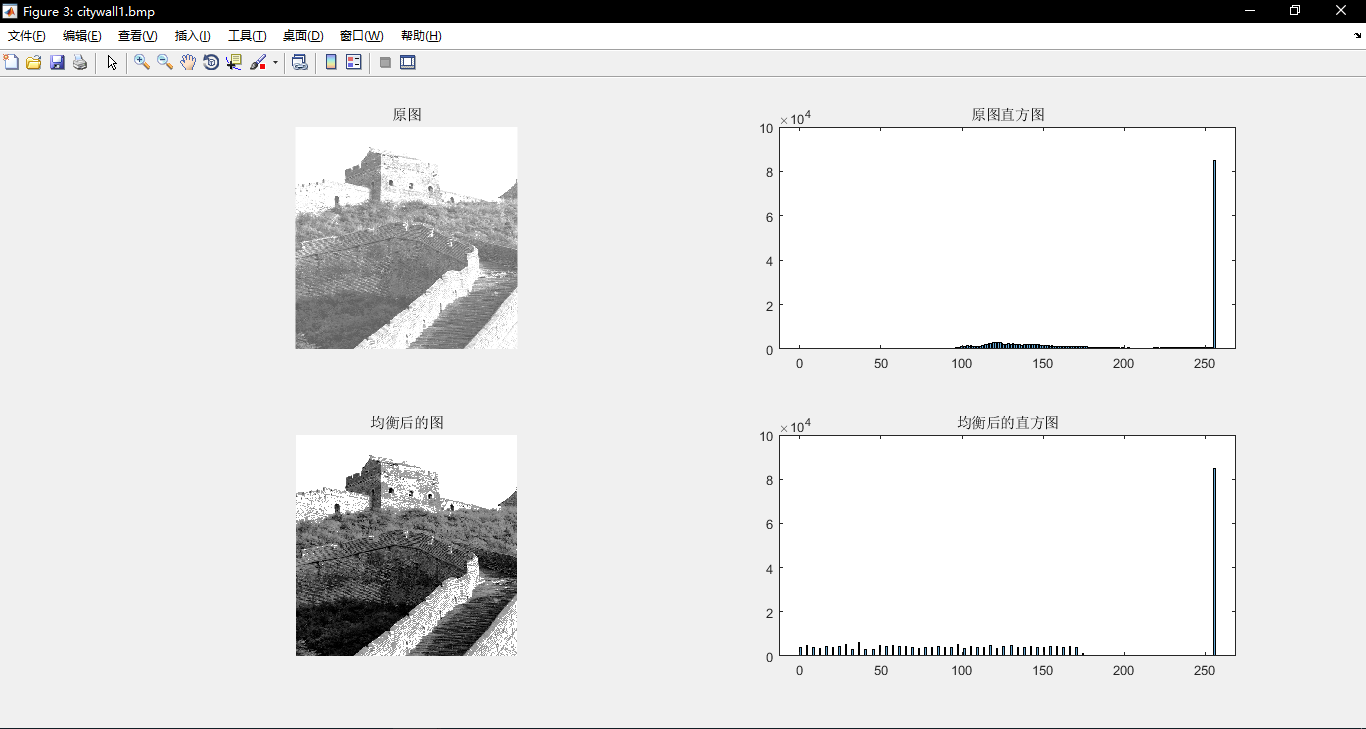


累积分布函数最小值cdfmin为最小的出现次数, M和N分别代表了图像的长宽像素个数，而L则是灰度级数。

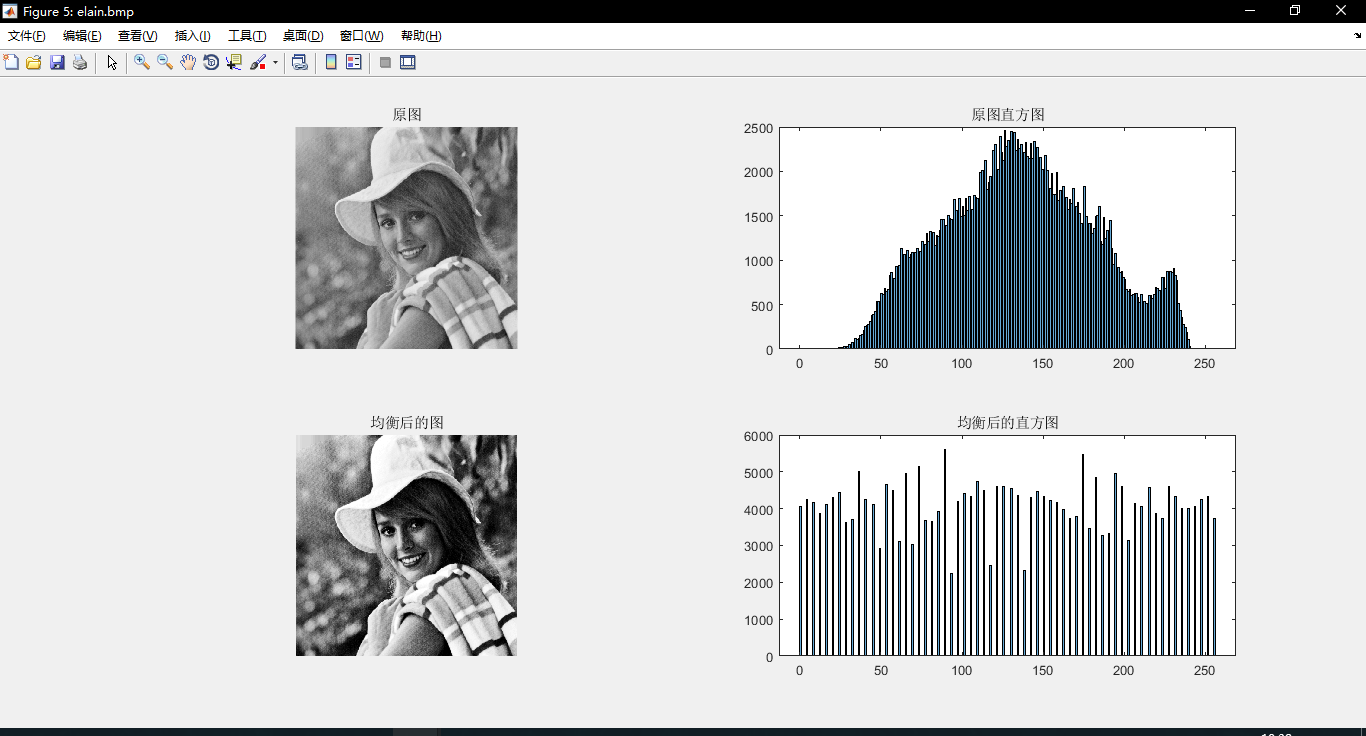
我编写了自己的直方图代码，同时也对比调用了Matlab的现有函数，发现两者几乎一致，这也验证了我代码的正确性

**结果展示：**

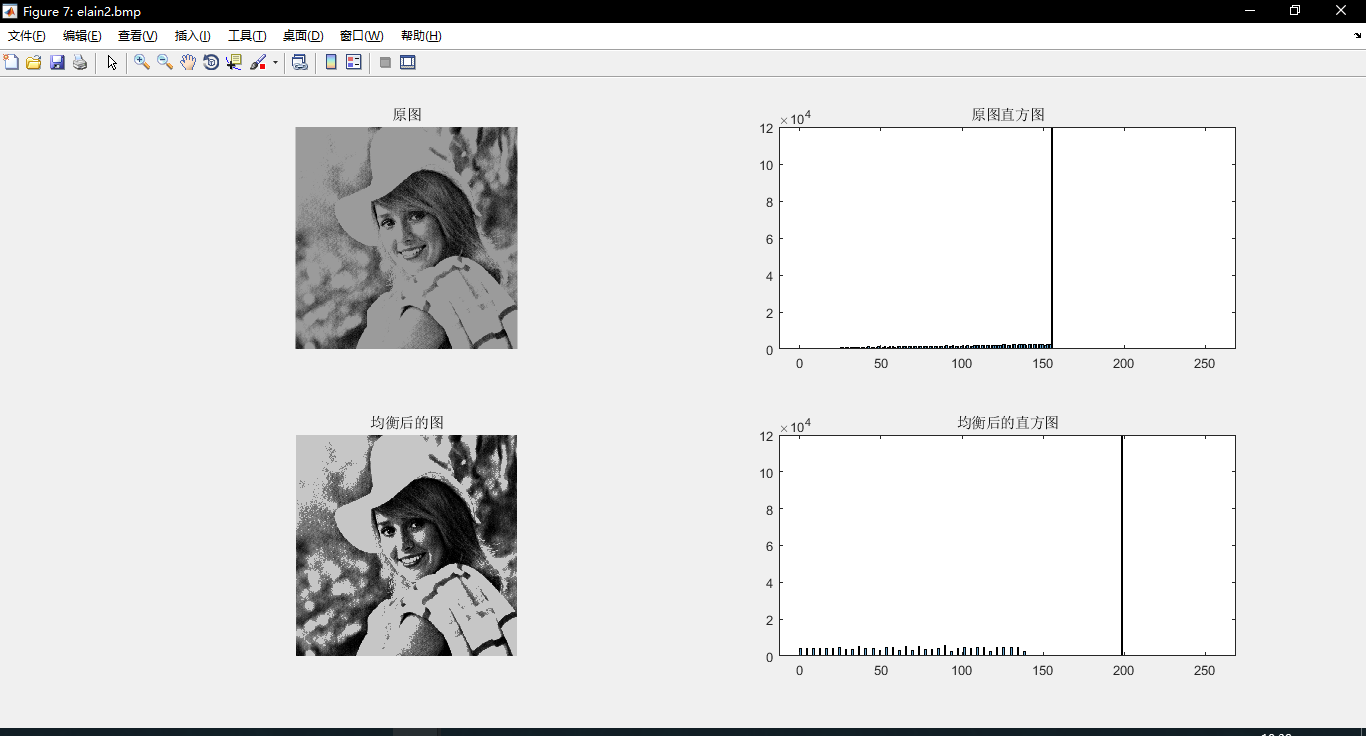


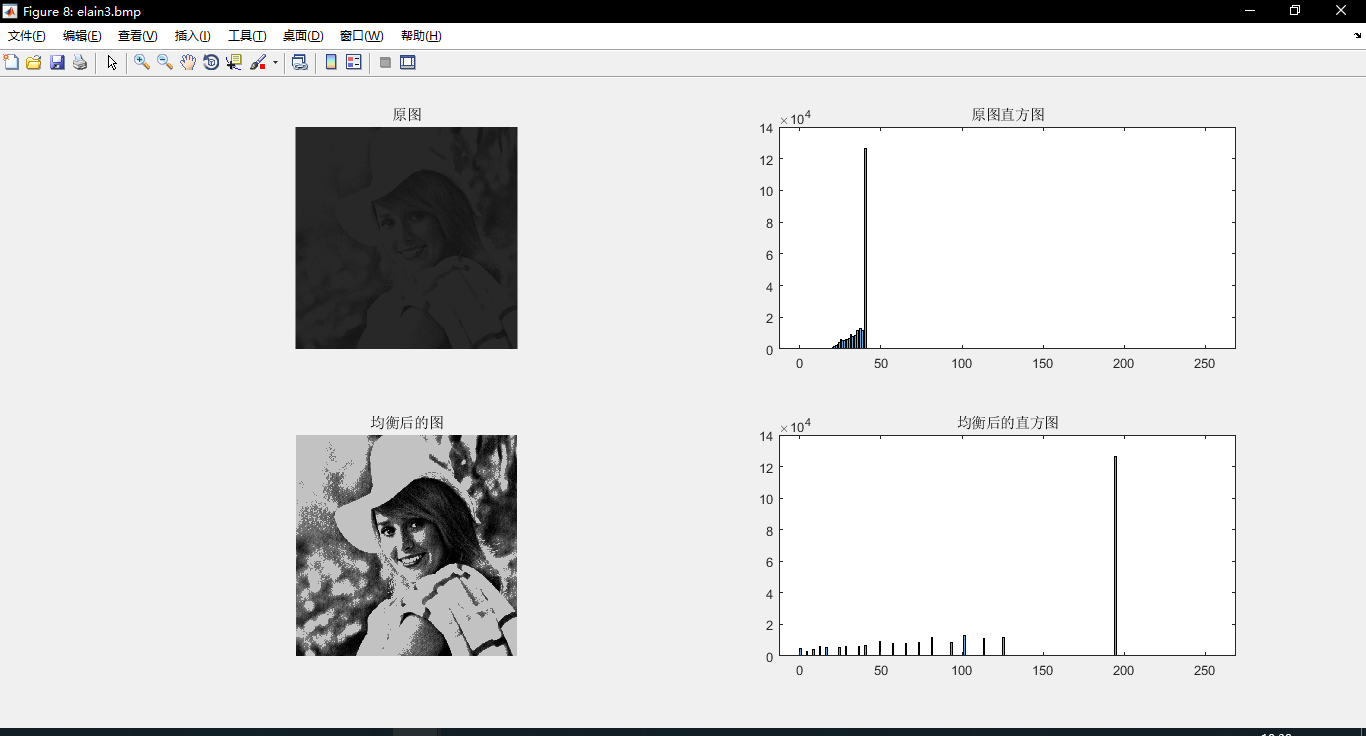


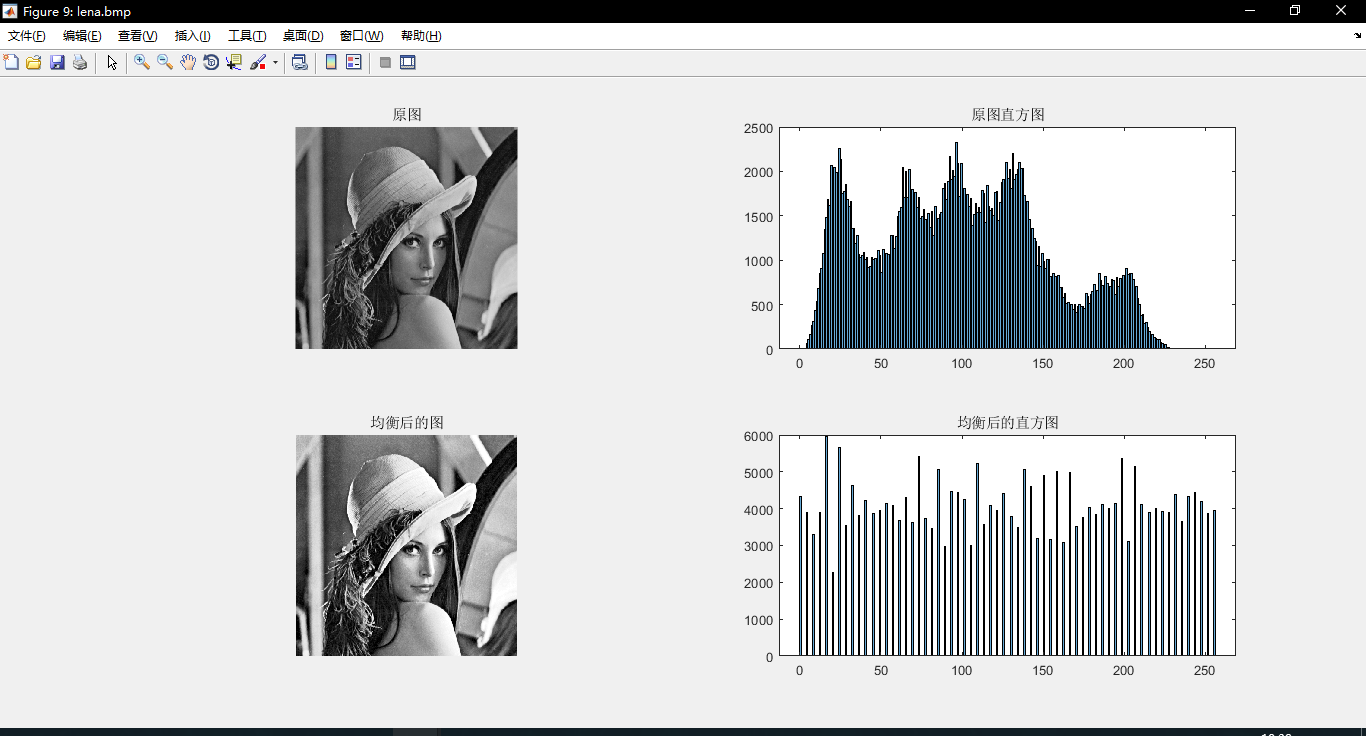


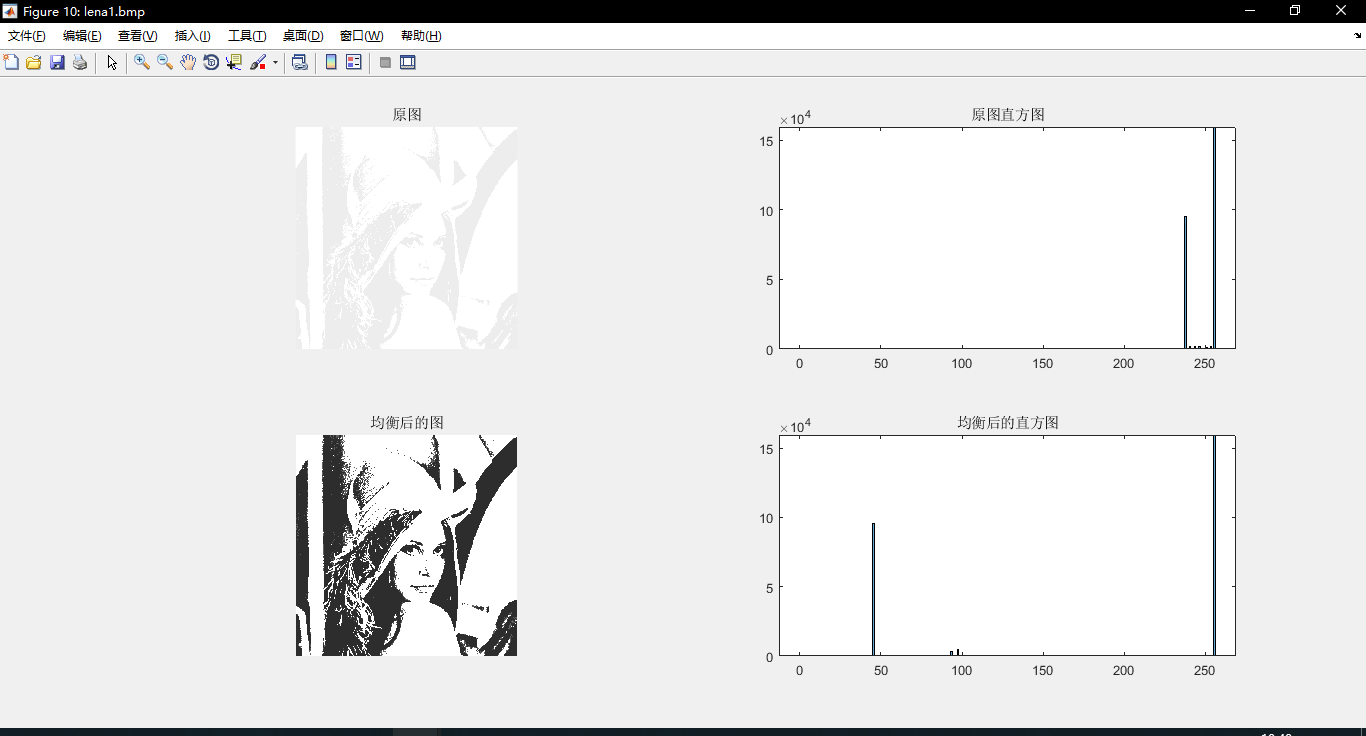


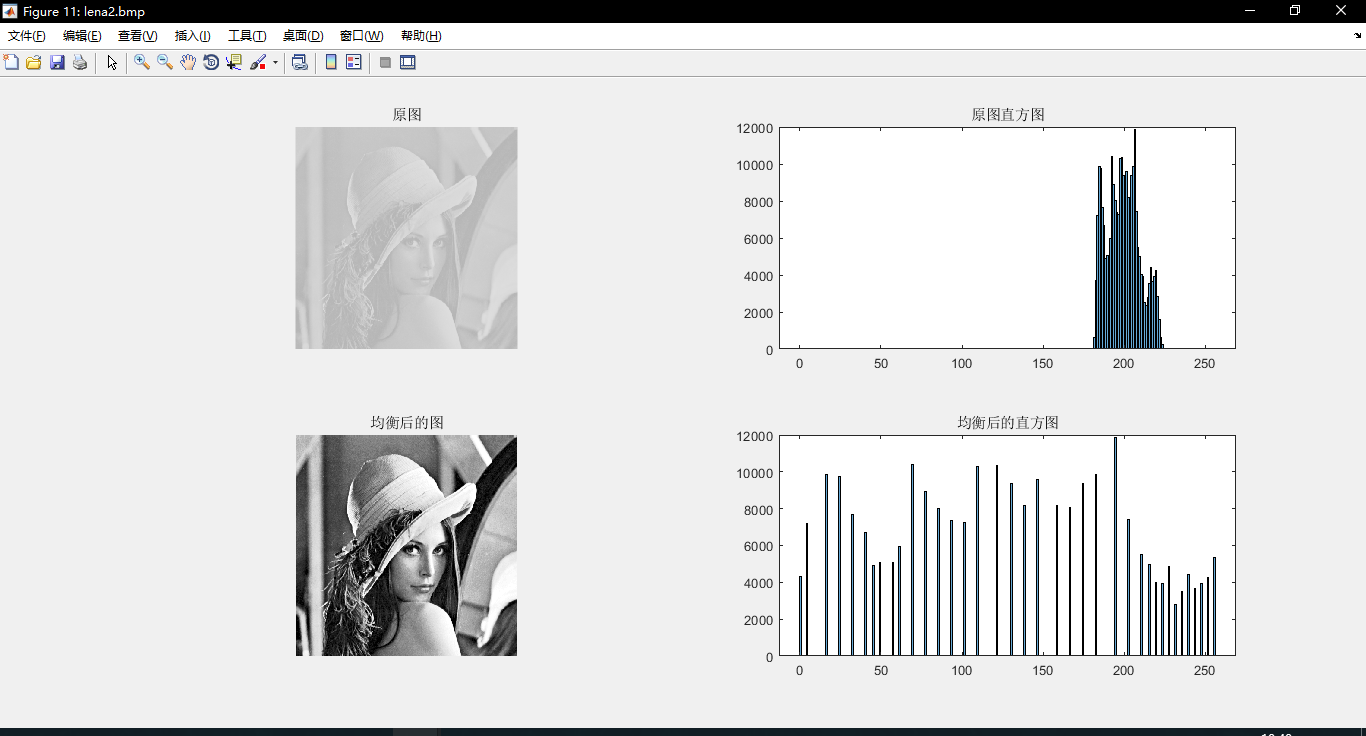


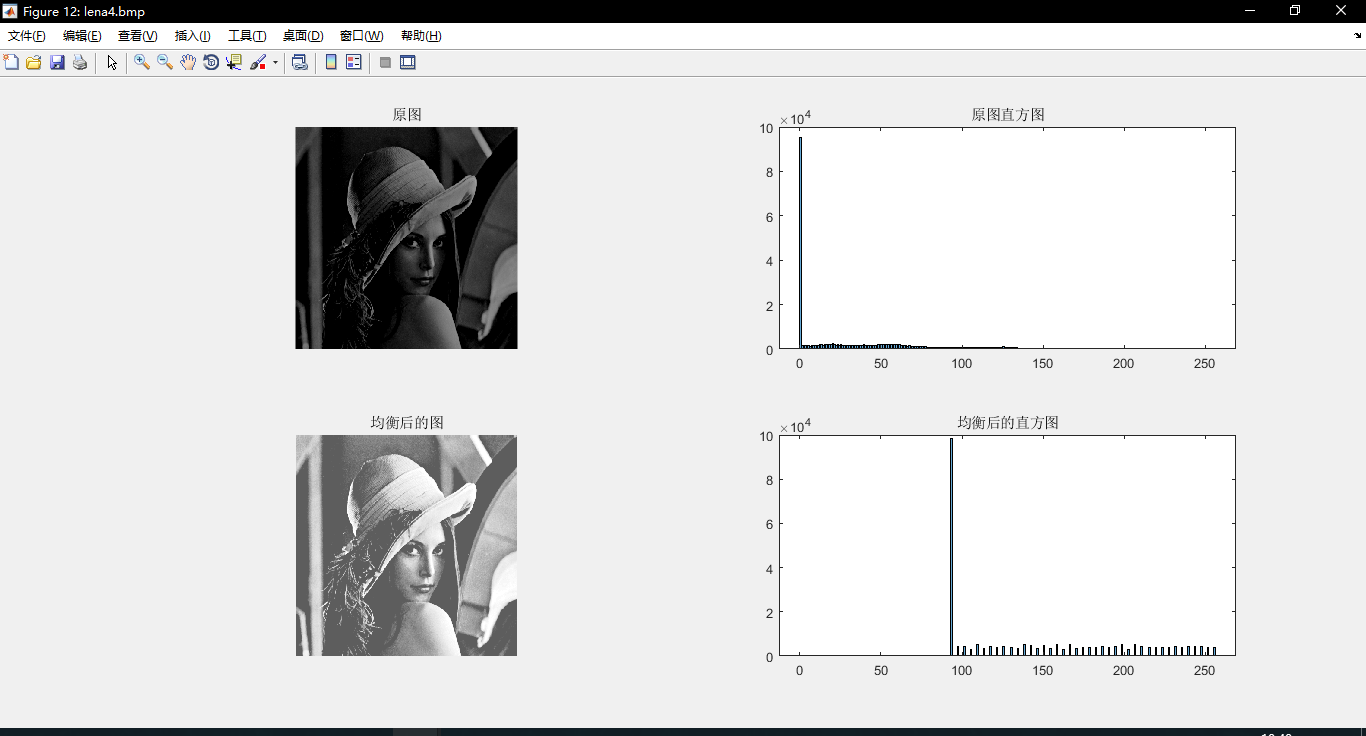


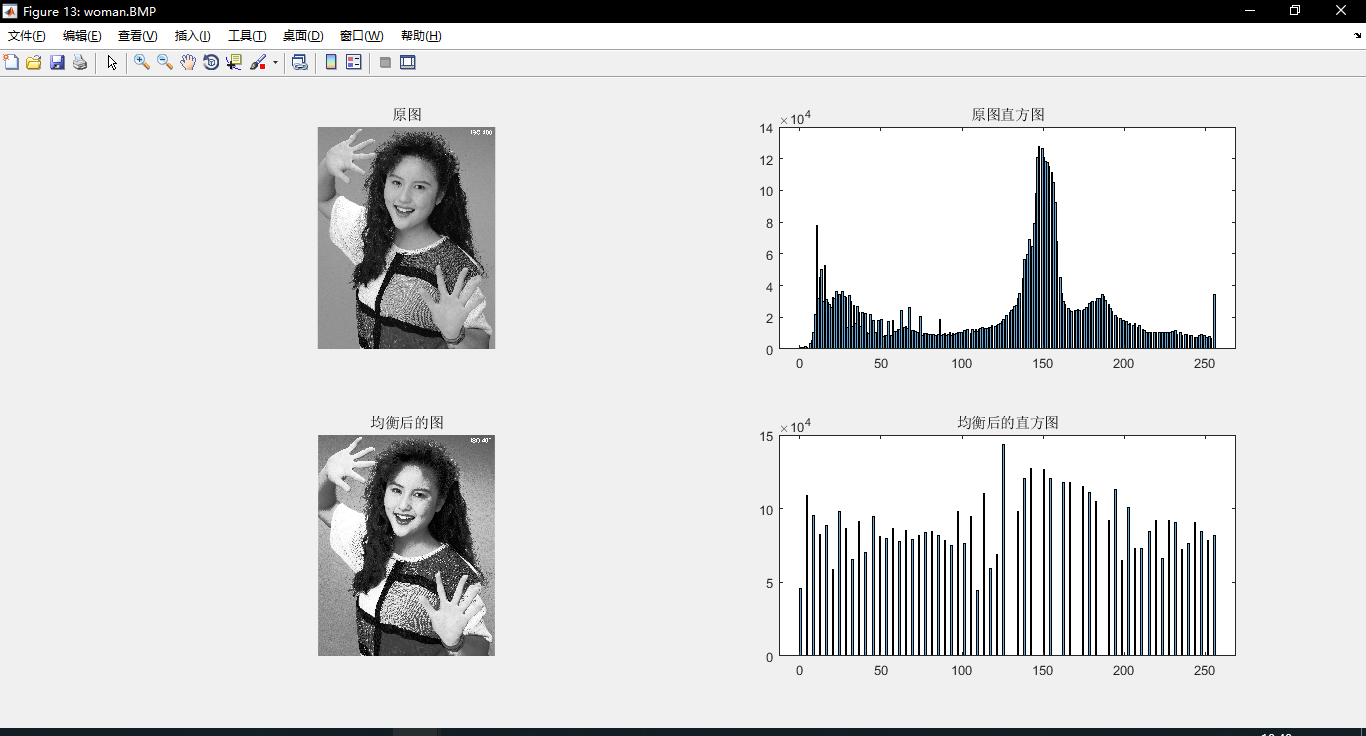












**结果讨论：**

通过以上对比可以看出，直方图均衡后，黑白的对比度加大，且原图的质量越好，均衡后的图片与原图的差异越小，那些变暗或者变亮了的图片在进行直方图均衡后黑白还原较好，与最原始的图相似。通过直方图显示的效果也可以很好地看出变化的过程。

## 任务3—直方图匹配

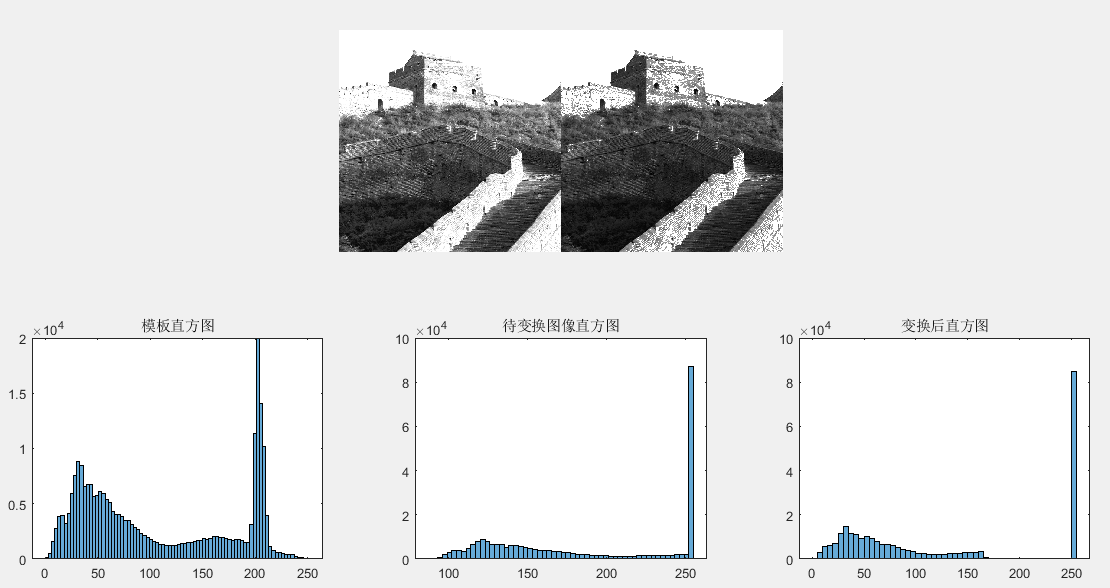
**实验介绍：**

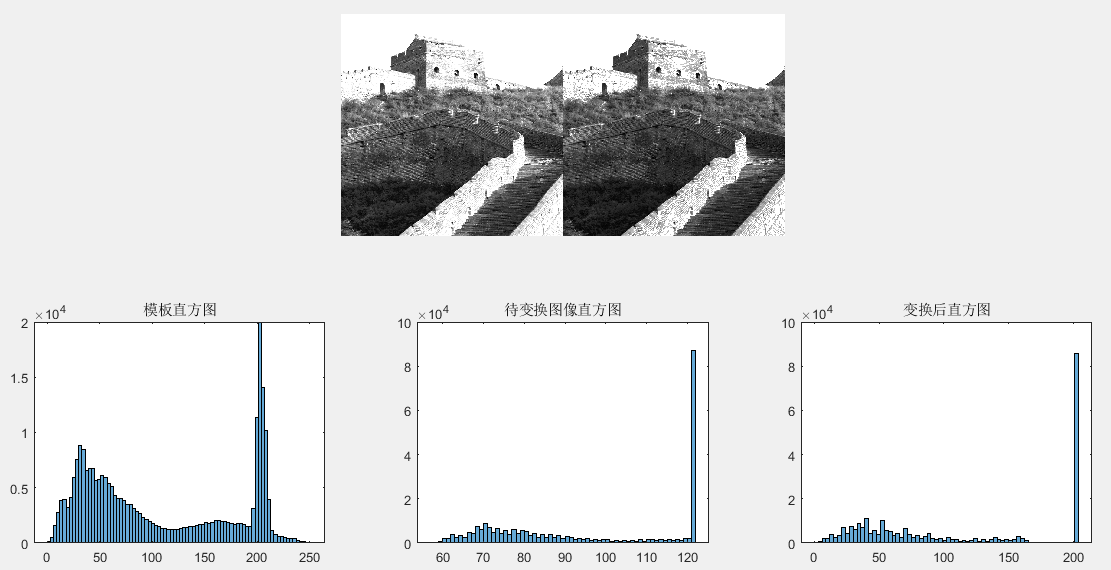
直方图匹配是使一幅图像的直方图变成规定形状的直方图而进行的图形增强办法，将图像直方图以标准图像的直方图为标准变换，使两图片的直方图相同和近似，从而使两幅图像具有类似的色调的色差。

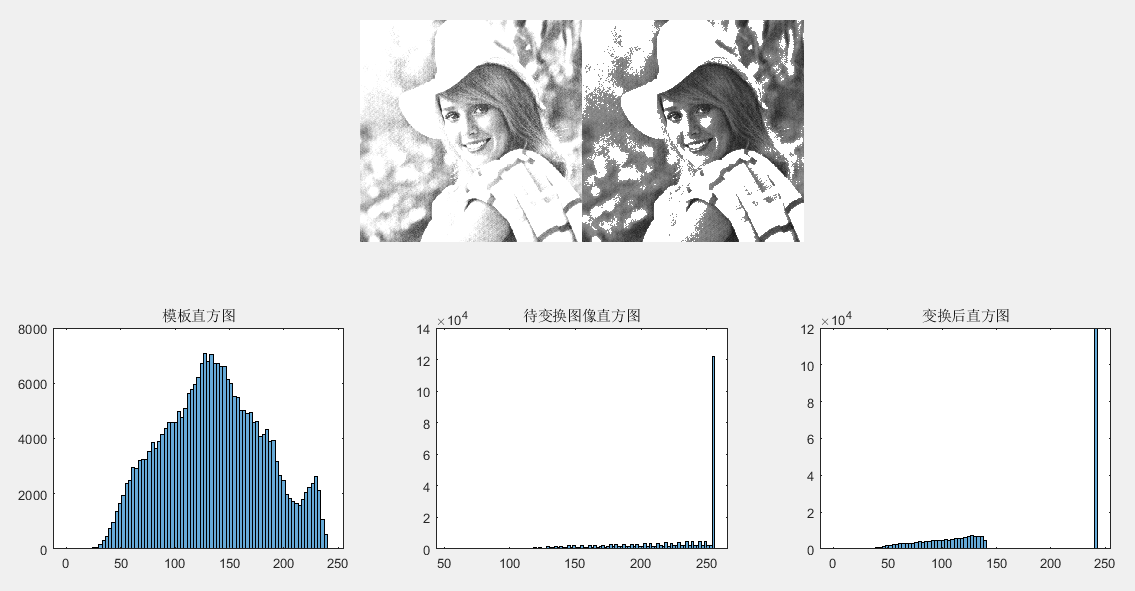
实际处理工作，选择模板直方图我们当然是选择名称没有数字的原图作为模板，用前面的获取直方图的函数获取模板直方图，然后这里我没有自己写函数来实现，而是直接调用的Matlab现有函数histeq来进行的实验。

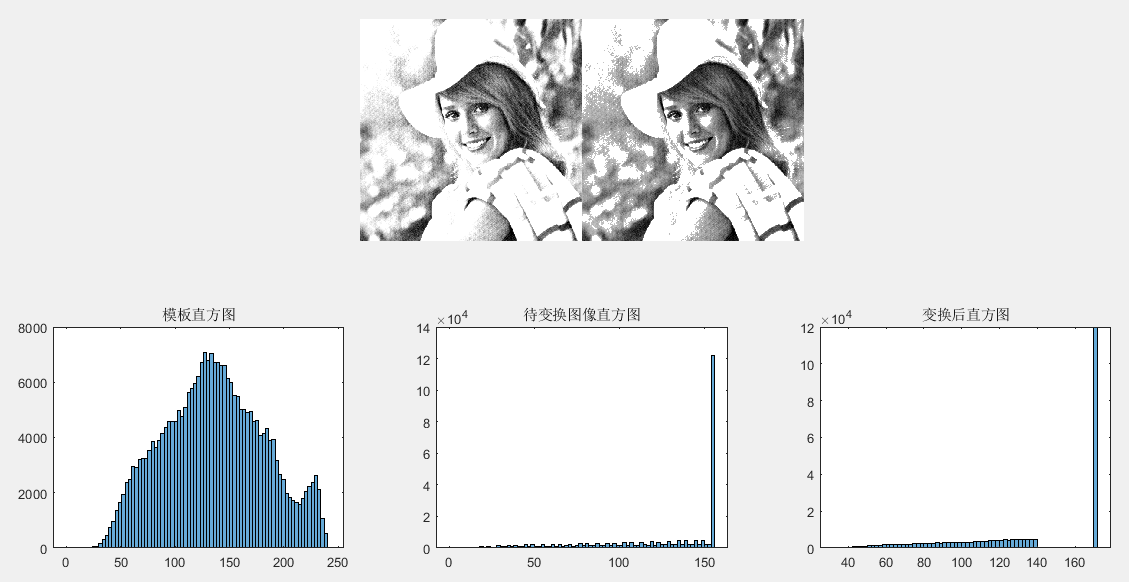
同样我也是用subplot将一次变换的所有展示放在了同一张图中。

**结果展示：**

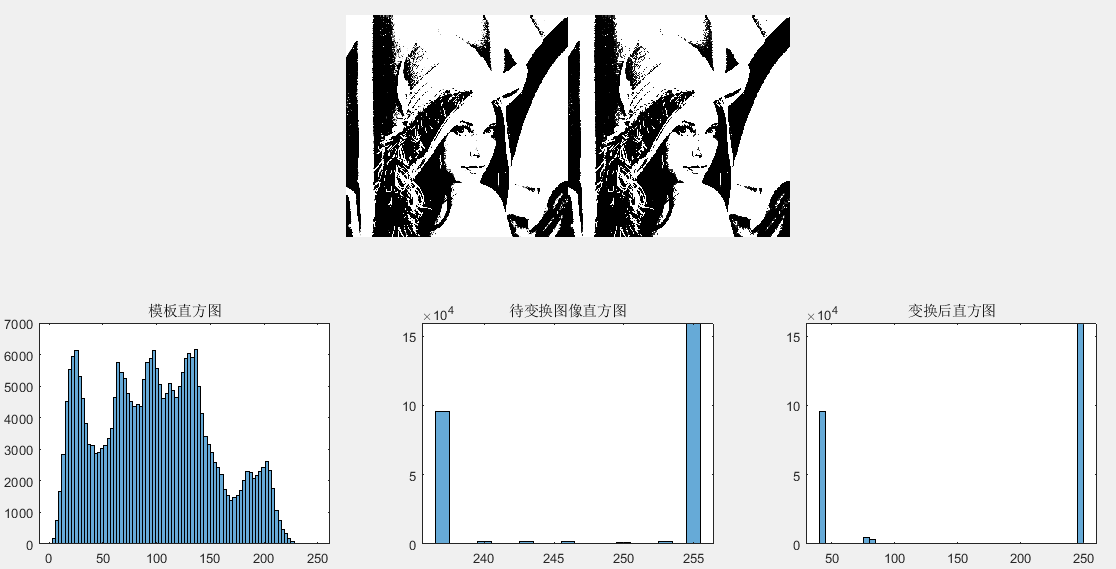


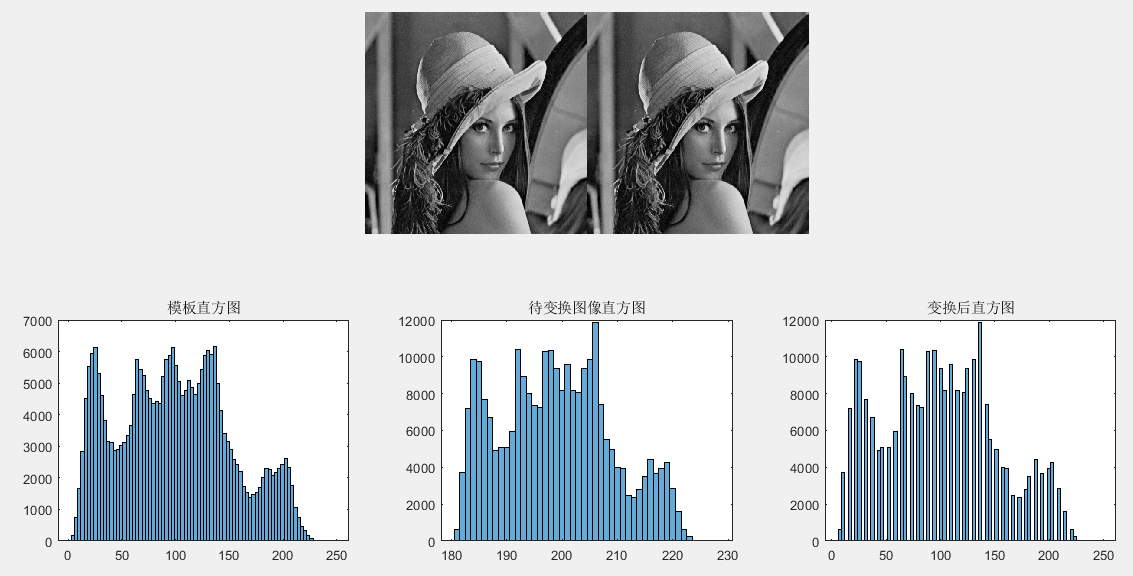


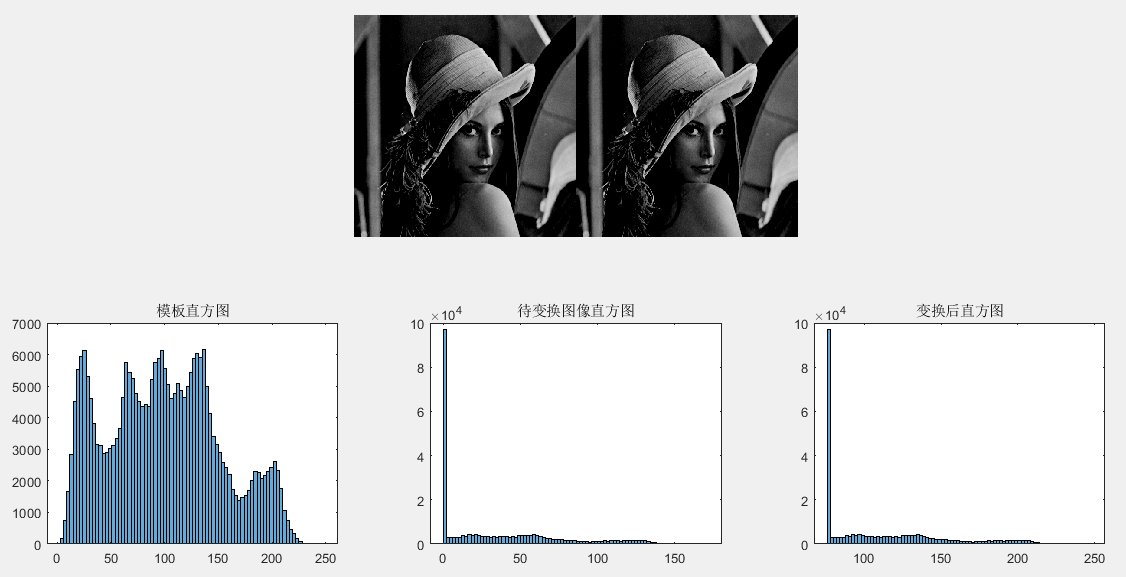












**结果讨论：**

通过对比可以发现待变换的图像直方图与模板差异很大，在进行直方图匹配后，并没有和模板直方图很像，而只是在趋势上达到了一致。图像的实际效果是往好的方向改进，所以也是进行了图像增强。

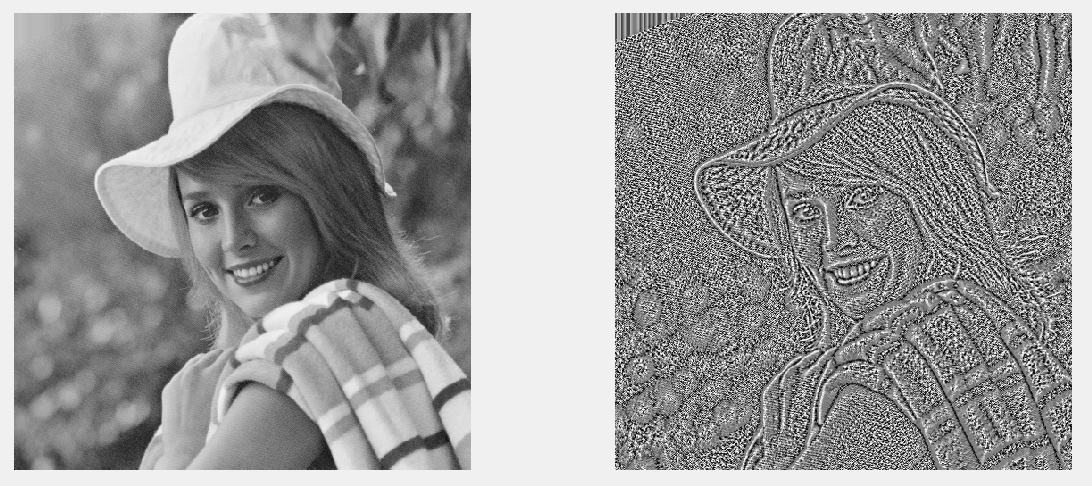
## 任务4—局部直方图均衡

**实验介绍：**

局部直方图均衡，顾名思义就是在全局的直方图均衡的基础上，以一个小块一个小块进行直方图的均衡，比如这次的题目就是说的以7\*7的块进行，那首先要把原图先扩展一下，原图是512\*512，那么边界就还要阔宽3，变成了518\*518的图像，然后从（1，1）开始对一个7\*7的矩阵进行直方图均衡变换，然后将最中间的（4，4）值付给一个新的矩阵，这个新形成的矩阵就是最后的结果。

注意这个局部直方图均衡在Matlab中是没有现有函数的。

**结果展示：**





**结果讨论：**

局部直方图均衡在这两个图片的处理中，图片的细节被消除，而对比度显著提高了。图像的显示更加特别。

## 任务5—直方图分割

**实验介绍：**

直方图分割的实质是利用图像的灰度值直方图信息获得用于分割的阈值，将图像的灰度级划分为几个部分，认为属于同一部份的像素是同一个物体。有多种进行直方图分割的办法，最简单的一种就是阈值法，这个方法适用于直方图呈现出双峰，那么阈值就可以选取两个峰中间的谷所在的灰度值，大于该阈值的重新赋值为1，小于它的赋值为0；本次实验中elain的图像就可以使用该方法。

还有一种较为复杂的方法，成为迭代法，它适用于没有这种明显双峰的图像，下面给出它的算法：

1. 选择阈值T，通常可以选择图像的平均灰度值来作为初始值
2. 通过初始阈值T，把图像的平均灰度值分为两组，R1和R2
3. 计算两组平均灰度值u1和u2
4. 重新选择阈值T，新的T=(u1+u2)/2；循环做第二步到第四步，一直到两组的平均灰度值不再发生变化，那么我们获得了所需要的阈值。

**结果展示：**

我先用的简单的阈值法处理的，我选定的两个阈值分别为150和100。



这是我用迭代法求出的结果如下，两个阈值分别为：137和101；





**结果讨论：**

在面对简单阈值划分时，用最简单的人工估计就可以解决，但对于稍微复杂一点的图片就没那么容易了。使用迭代法可以很容易地用循环找到一个最佳阈值，这样的方法我们应该多家掌握。

## 代码：

% 1.把附件图像的直方图画出；

function y = myhist(img)

y = zeros(256,1);

[m,n] = size(img);

for i = 1:m

for j = 1:n

y(img(i,j)+1)=y(img(i,j)+1)+1;

end

end

clear;clc;

file\_structure = dir('\*.bmp');

for i = 1:size(file\_structure,1)

subplot(3,4,i);

[picture{i},map{i}] = imread(file\_structure(i).name); %把图片的调色盘也都进去

p{i}=ind2gray(picture{i},map{i});

edges = [0:256];

histogram(p{1,i},edges);

title(sprintf(file\_structure(i).name));

end

% 2.把所有图像进行直方图均衡；输出均衡后的图像和源图像进行比对；分析改善内容；

function y = myhist\_equ(img)

a = myhist(img);

[m,n] = size(img);

a\_p = a / (m \* n);

b = zeros(1,256);

for i = 1:256

t = 0;

for j = 1:i

t = t + a\_p(j);

end

b(i) = floor(255 \* t);

end

y = uint8(zeros(m,n));

for i = 1:m

for j = 1:n

y(i,j) = uint8(b(img(i,j)+1));

end

end

clc;clear;

file\_structure = dir('\*.bmp'); %加载根目录下的.bmp格式的所有文件

edges = [0:256]; %设置直方图的显示边界

for i = 1:size(file\_structure,1)

figure(i); %画第i个图

figure('Name',sprintf(file\_structure(i).name));

[picture{i},map{i}] = imread(file\_structure(i).name); %读图片，包括了调色盘

p{i}=ind2gray(picture{i},map{i}); %将调色盘的效果加载上去，产生真正效果的图

subplot(2,2,1);

imshow(p{i}); %显示原图

title(sprintf('原图'));

subplot(2,2,2);

histogram(p{i},edges); %显示原图的直方图

title(sprintf('原图直方图'));

pp=histeq(p{1,i}); %作直方图均衡变换

subplot(2,2,3);

imshow(pp); %显示均衡后的直方图

title(sprintf('均衡后的图'));

subplot(2,2,4);

histogram(pp,edges); %显示均衡后的直方图

title(sprintf('均衡后的直方图'));

end

% 3.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；

clc;clear;

picture=imread('lena.bmp');

shist=imhist(picture); %获取模板直方图

[picture1,map1]=imread('lena4.bmp');

p1=ind2gray(picture1,map1); %得到真是的图像数据

pp=histeq(p1,shist); %调用函数进行直方图均衡

subplot(2,3,[1,3]);

imshowpair(p1,pp,'montage'); %显示原图和匹配后的图像

subplot(2,3,4);

histogram(picture);

title(sprintf('模板直方图'));

subplot(2,3,5);

histogram(p1);

title(sprintf('待变换图像直方图'));

subplot(2,3,6);

histogram(pp);

title(sprintf('变换后直方图'));

% 4.对elain和lena图像进行7\*7的局部直方图增强；

tic

picture1 = imread('elain.bmp');

subplot(2,2,1);

imshow(picture1);

p1 = imresize(picture1,[518,518],'bicubic'); %双三次插值;

for i=1:512

for j=1:512

block = p1(i:i+6,j:j+6 ); %动态的7\*7矩阵来储存变量

pp1 = histeq(block); %对这个块进行直方图均衡处理

ppp1(i,j) = pp1(4,4); %再把block中间的那个值赋给新的图像矩阵

end

end

subplot(2,2,2);

imshow(ppp1);

picture2 = imread('lena.bmp');

subplot(2,2,3);

imshow(picture2);

p2 = imresize(picture2,[518,518],'bicubic'); %双三次插值;

for i=1:512

for j=1:512

block = p2(i:i+6,j:j+6 ); %动态的7\*7矩阵来储存变量

pp2 = histeq(block); %对这个块进行直方图均衡处理

ppp2(i,j) = pp2(4,4); %再把block中间的那个值赋给新的图像矩阵

end

end

subplot(2,2,4);

imshow(ppp2);

toc

% 5.利用直方图对图像elain和woman进行分割

%采用最简单阈值法的代码

clc;clear;

picture1=imread('elain.bmp');

subplot(2,2,1);

imshow(picture1);

threshold1=150;

[m1,n1]=size(picture1);

pp=zeros(m1,n1);

for i=1:m1

for j=1:n1

if(picture1(i,j)>threshold1)

pp(i,j)=1;

else

pp(i,j)=0;

end

end

end

subplot(2,2,2);

imshow(pp);

picture2=imread('woman.bmp');

subplot(2,2,3);

imshow(picture2);

threshold2=100;

[m2,n2]=size(picture2);

ppp=zeros(m2,n2);

for i=1:m2

for j=1:n2

if(picture2(i,j)>threshold2)

ppp(i,j)=1;

else

ppp(i,j)=0;

end

end

end

subplot(2,2,4);

imshow(ppp);

%采用迭代法的代码

clc;clear;

picture = imread('woman.bmp');

subplot(1,2,1);

imshow(picture); %显示图片

title('原图woman.bmp');

picture = double(picture); %转换为double类型为了后面累加计算

[m,n] = size(picture); %获得图片的大小

T = 90; %设定初始阈值

T1 = 0;T2 = 255;

while(1)

sum1 = 0;sum2 = 0; %为了累加每一类的灰度值

count1 = 0;count2 = 0; %为了累加每一类的数量

for i = 1:m

for j = 1:n

if picture(i,j) > T

sum1 = sum1 + picture(i,j);

count1 = count1+1;

else

sum2 = sum2 + picture(i,j);

count2 = count2 + 1;

end

end

end

if abs(T1-T2) == abs(sum1/count1-sum2/count2) %循环的终止条件，不再变化，注意不是两者均值相等

break;

end

T1 = sum1/count1; %计算第一类平均灰度值

T2 = sum2/count2; %计算第二类平均灰度值

T = (T1 + T2)/2; %更新阈值T

end

fprintf('阈值为： %8.5f\n',T)

J = zeros(m,n);

for i = 1:m

for j = 1:n

if picture(i,j)>T

J(i,j) = 1;

else

J(i,j) = 0;

end

end

end

subplot(1,2,2);

imshow(J)

title('图像分割woman.bmp');

## 参考文献

[1] 冈萨雷斯.数字图像处理（第三版）北京：电子工业出版社，2011

[2] 维基百科