中山大學
SUN YAT-SEN UNIVERSITY

本科生实验报告

学生姓名：　丁晓琪

学生学号：　22336057

专业名称：　计科

一：实验要求

1. 直方图均衡化编写一个计算图像直方图的程序，实现3.3.1节中讨论的直方图均衡化技术，下载图(a)，对其进行直方图均衡化。（结果应包括原始图像、其直方图的图、直方图均衡转换函数的图、增强图像及其直方图的图）
2. 利用拉普拉斯算子增强根据式(3.54)相关的拉普拉斯增强技术，利用图3.45(d)所示的拉普拉斯核，下载图(b)，对其进行拉普拉斯锐化
3. 提取并展示灰度图像的8个位平面；随意修改最低位平面值，观测修改后图与原图差别。

二：实验过程

1. 直方图均衡

(1). 理论背景：

- 将像素灰度值r_k通过函数$T(r)$变化为灰度值s_k，使得变换后的图像上灰度值分布更加均匀，提高对比度

$$s_k = (L-1)\sum_{j=0}^{k}\frac{n_k}{MN}$$

$L-1$：灰度值范围的最大值

n_k：灰度值为k的像素个数

MN：M为图像的宽，N为图像的高，MN为图像的总的像素数

- 如果是连续灰度直方图均衡后，所有灰度值在图像中统计的频率相同都为$\frac{1}{L-1}$，，但是如果是离散灰度直方图均衡后，图像中不同灰度值的频率值无法完全相同

(2). 代码实现：

1. 提取出图像的灰度值矩阵后，计算得到归一化后的直方图
   $p(r_k)=\frac{n_k}{MN}$: 灰度值k的频率为图中所有灰度值为k的像素在总像素值中的占比

```python
#归一化处理，计算直方图
def calculate_histogram(gray_matrix):
    # gray_matrix:图像的灰度值矩阵
    # histogram：为长度256的list，histogram[i]为灰度值i在图像中的频率
    histogram=[0]*256
    width=gray_matrix.shape[1]
    height=gray_matrix.shape[0]
    sum=width*height
    for x in range(0,height):
        for y in range(0,width):
            gray_value=gray_matrix[x][y]
            histogram[int(gray_value)]+=1
    for i in range(0,len(histogram)):
        histogram[i]/=sum
    return histogram
```

2. 直方图均衡转换函数
   - 计算累积分布cdf：每个灰度级之前的所有灰度级的累计像素频率
   - 转换图像 $s_k=(L-1)\sum_{i=0}^{k}cdf[i]$ (原图像的灰度值k在新图像上转换为s_k)

```python
#直方图均衡
def histogram_equalization(histogram,gray_matrix):
    #输入：histogram，原图像的直方图
    #输入：gray_matrix:原图像的灰度值矩阵
    #输出：after_matrix：直方图均衡后的图形灰度值矩阵
    #1. 计算累积分布
    cdf=[0]*256
    cdf[0]=histogram[0] #初始化
    width=gray_matrix.shape[1]
```

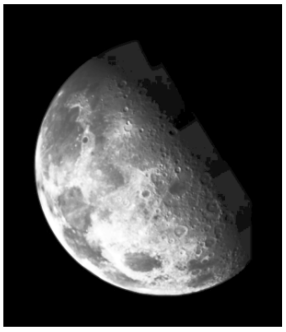
```
    height=gray_matrix.shape[0]
    for i in range(1,len(cdf)):
        cdf[i]=cdf[i-1]+histogram[i]
    #2.计算直方图均衡后的灰度矩阵
    after_matrix=np.zeros((height,width))
    for x in range(0,height):
        for y in range(0,width):
                gray_values=gray_matrix[x][y]
                sk=(255)*cdf[gray_values]
                after_matrix[x][y]=int(sk)

    return after_matrix
```

(3). 实验结果

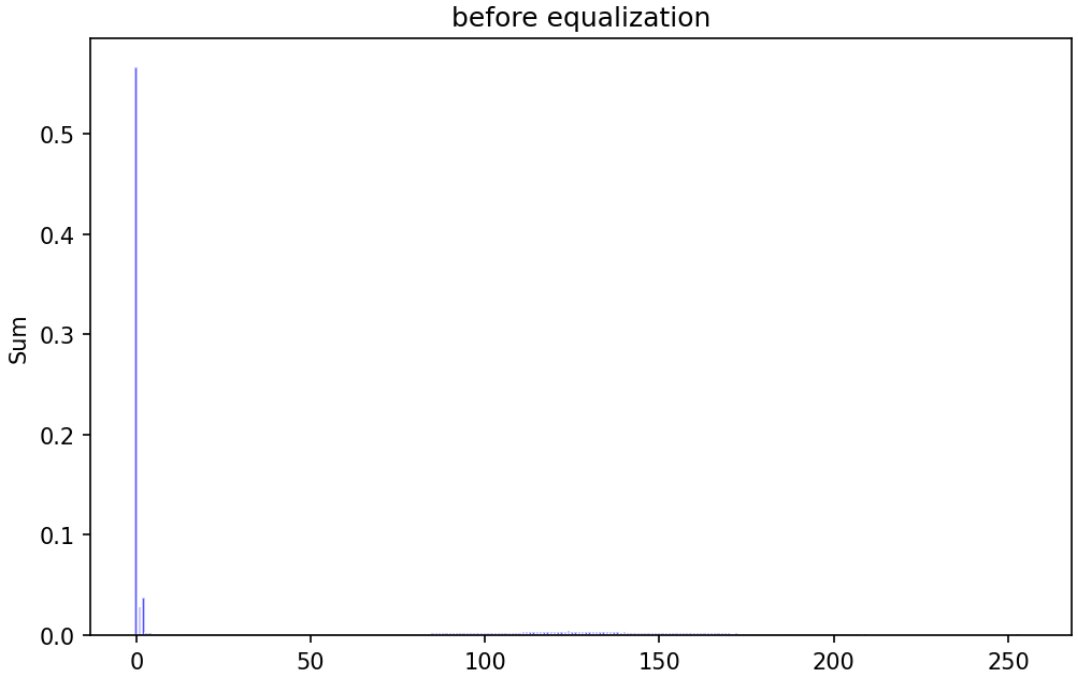
- 对比均衡前后的图像：显然直方图均衡后的图像对比度更高，细节更多，边界也比较清晰

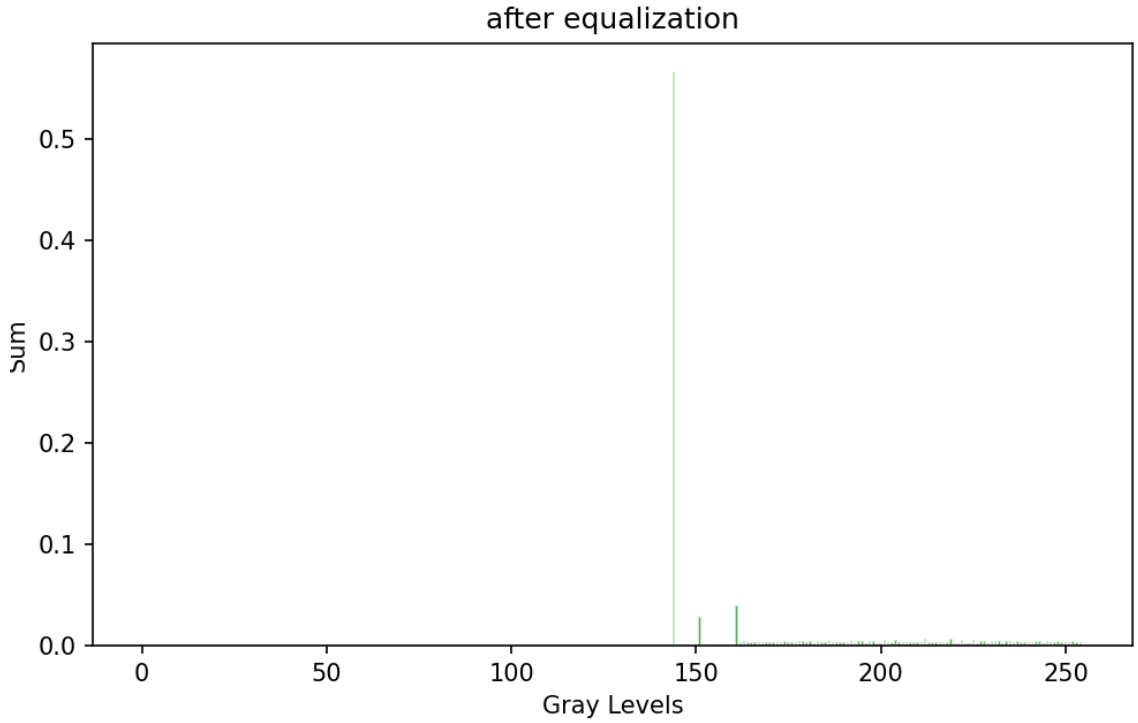
before equalization

after equalization



- 对比均衡前后的直方图：
  - 由于灰度值为0的占比实在太高，导致直方图分布极端，而且由于$s_k = (L-1)\sum_{i=0}^{k} cdf[i]$，如果cdf[0]过大，后面的灰度值占比过小，则在转换计算中可能导致灰度值合并。sk取整，淹没了一些灰度值的转换，导致转换后一些灰度值在转换后出现的频率为0





- 打印出直方图的值观察：转换后的图像在灰度值大于150后的部分分布比较均匀

```
before [0.566594026406871, 0.02809335341622869, 0.03763139341110114, 0.002379502627868222, 0.0016303999487245226, 0.0015823291885655685, 0.0013339635944100973, 0.0011657159338544634, 0.001185745
4172541982, 0.0011336687604153314, 0.0011096333803358545, 0.0009934623766183822, 0.0009493975131393411, 0.0010335213434175105, 0.0010575567234969877, 0.0010054800666581206, 0.00088530316626073
58, 0.0009333739264196898, 0.0008933149596205615, 0.0009213562363799513, 0.0009493975131393411, 0.0009894564799384695, 0.0009493975131393411, 0.0009333739264196898, 0.0008172029227022176, 0.00
10695744135367261, 0.0009854505832585566, 0.0008853031662607358, 0.0009213562363799513, 0.0009734328932188181, 0.0009654210998589924, 0.0010815921035764646, 0.001045539033457249, 0.00093737982
30996026, 0.001025509550057685, 0.000997468273298295, 0.0009894564799384695, 0.0009734328932188181, 0.0010094859633380335, 0.0009894564799384695, 0.0010535508268170748, 0.0009333739264196898,
0.0010695744135367261, 0.001001474169978208, 0.0009854505832585566, 0.0009934623766183822, 0.0010695744135367261, 0.0010896038969362902, 0.0011496923471349828, 0.0010295154467375978, 0.0010735
803102166388, 0.00111764517369568, 0.0010815921035764646, 0.0010735803102166388, 0.0011737277272144597, 0.0011977631072939367, 0.0012538456608127164, 0.0012979105242917576, 0.00131794000769132
16, 0.0013259518010511472, 0.00132995769773106, 0.0014100756313293167, 0.0013459812844507115, 0.0014701640815280093, 0.0014421228047686195, 0.0014020638379694912, 0.001426099218048968, 0.00143
41110114087937, 0.0015302525317267017, 0.0015903409819253943, 0.001498205358287399, 0.0015622997051660044, 0.0015302525317267017, 0.0015382643250865273, 0.0014581463914882706, 0.00145814639148
82706, 0.001430105114728881, 0.00132995769773106, 0.0012658633508524548, 0.0012538456608127164, 0.0013740225612101013, 0.0012898987309319318, 0.0012939046276118447, 0.0012778810408921933, 0.00
15262466350467888, 0.0018387065760799898, 0.0017225355723625177, 0.0017585886424817331, 0.0017906358159210359, 0.0017946417126009485, 0.0017666004358415587, 0.001878765542879118, 0.00198692475
32367643, 0.0018667478528393796, 0.001998942443276503, 0.002047013203435457, 0.002095083963594411, 0.0023194141776695294, 0.0021151134469939753, 0.0022793552108704013, 0.002331431867709268, 0.
002259325727470837, 0.0023154082809896167, 0.002327425971029355, 0.002375496731188309, 0.0023234200743494425, 0.0022953787975900524, 0.0025717856685040378, 0.0024435969747468272, 0.00259982694
5263428, 0.0026078387386232536, 0.002952345853095757, 0.002952345853095757, 0.003048487373413665, 0.003048487373413665, 0.0030124343032944495, 0.003332906037687476, 0.003208723240610178, 0.003
240770414049481, 0.0034210357646455585, 0.003489136008204076, 0.003489136008204076, 0.003729489808998846, 0.003521183181643379, 0.003757531085758236, 0.003549224458402769, 0.003481124214844251
, 0.003645365978720677, 0.0033929944878861684, 0.0035772657351621586, 0.003328900141007563, 0.003429047558005384, 0.0033248942443276502, 0.003164658377131137, 0.0029843930265350594, 0.00301644
0199974362, 0.0030885463402127934, 0.0027520510191001153, 0.0027560569157800284, 0.0026078387386232536, 0.0027159979489809, 0.0026078387386232536, 0.002527720805024997, 0.002595821048583515, 0
.0025116972183053454, 0.0026519036021022944, 0.0023714908345083964, 0.002211254967311883, 0.002267337520830663, 0.0021752018971926677, 0.0022913729009101397, 0.002119119343673888, 0.0022993846
942699655, 0.002191225483912319, 0.0023194141776695294, 0.002111107550314062, 0.0024395910780669145, 0.0021752018971926677, 0.0022993846942699655, 0.0020630367901551083, 0.0020349955133957187,
 0.0022913729009101397, 0.002095083963594411, 0.0019588834764773747, 0.0018587360594795538, 0.00199493654659659, 0.0017706063325214716, 0.0018707537495192923, 0.0016544353288039995, 0.00175858
86424817331, 0.001642417638764261, 0.0016023586719651327, 0.0016584412254839122, 0.0015382643250865273, 0.0015102230483271376, 0.0014140815280092296, 0.0013379694910908859, 0.00131393411101140
87, 0.0014421228047686195, 0.00135799897449045, 0.0012097807973336751, 0.0013059223176515831, 0.0011376746570952443, 0.0011777336238943726, 0.0010535508268170748, 0.0010174977566978593, 0.0010
775862068965517, 0.0009453916164594283, 0.0009654210998589924, 0.0009213562363799513, 0.0009413857197795155, 0.0009213562363799513, 0.0009373798230996026, 0.0009013267529803872, 0.000845244199
4616075, 0.0008412383027816947, 0.0007450967824637867, 0.0007891616459428278, 0.0007571144725035252, 0.0006729906422253557, 0.0007611203691834381, 0.0007290731957441353, 0.0006689847455454429,
 0.0006088962953467505, 0.0006529611588257916, 0.000548807845148058, 0.0005608255351877965, 0.0005968786053070119, 0.0004927252916292783, 0.000548807845148058, 0.00036453659787206766, 0.000448
66042815023714, 0.0004166132547109345, 0.0004406486347904115, 0.0004406486347904115, 0.000476701704909627, 0.0004326368414305858, 0.0004686899115498013, 0.0003525189078323292, 0.00042462504807
076014, 0.00036053070119215487, 0.00036053070119215487, 0.0003525189078323292, 0.00032047173439302655, 0.0003164658377131137, 0.00029643635431354955, 0.0002924304576336367, 0.00033649532111267
787, 0.000308454044353288, 0.0003685424945519805, 0.00024435969747468276, 0.00021231252403538007, 0.00025637738751442123, 0.0002283361107550314, 0.00023634790411485707, 0.0002283361107550314,
0.00023234200743494423, 0.0003004422509934624, 0.0002403538007947699, 0.00024836559415459554, 0.0002002948339956416, 0.0002002948339956416, 0.0002043007306755544, 0.00018427124727599025, 0.000
1922830406358159, 0.0001642417638764261, 0.00014821817715677477, 0.0001762594539161646, 0.0002043007306755544, 0.00024435969747468276, 0.00022433021407511857, 0.00018427124727599025, 0.0003164
658377131137, 0.0008933149596205615, 0.0002844186642738111, 1.6023586719651327e-05]
after [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.566594026406871, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.02809335341622869, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.04001089603896936, 0.003212729137290091, 0.004819093705935137, 0.004086014613511088, 0.003881713882835534, 0.0037535
251890783235, 0.0037695487757979746, 0.003765542879118062, 0.004029932059992309, 0.003985867196513267, 0.003985867196513267, 0.004049961543391873, 0.003308870657607999, 0.0043023330342263815,
0.0034450711447250354, 0.0038696961927957955, 0.004065985130111524, 0.00425826817074734, 0.004262274067427253, 0.0031205935136520957, 0.004590757595180105, 0.002996410716574798, 0.004218209203
948212, 0.0038937315728752725, 0.0038616843994359698, 0.005087488783489296, 0.003549224458402769, 0.0035612421484425075, 0.0038656902961158824, 0.0038656902961158824, 0.004142097167029868, 0.0
04434527624663505, 0.0022793552108704013, 0.004590757595180105, 0.004642834252018972, 0.002375496731188309, 0.004618798871939495, 0.005015382643250865, 0.002599826945263428, 0.0026078387386232
536, 0.005904691706191514, 0.003048487373413665, 0.003048487373413665, 0.0063453403409819255, 0.003208723240610178, 0.003240770414049481, 0.0034210357646455585, 0.003489136008204076, 0.0034891
36008204076, 0.003729489808998846, 0.003521183181643379, 0.007306755544161005, 0.003481124214844251, 0.003645365978720677, 0.0033929944878861684, 0.0035772657351621586, 0.003328900141007563, 0
.003429047558005384, 0.006489552621458787, 0.0029843930265350594, 0.003016440199974362, 0.005840597359312909, 0.0027560569157800284, 0.0026078387386232536, 0.0053238366876041535, 0.00252772080
5024997, 0.0051075182668886, 0.005023394436610691, 0.002211254967311883, 0.00444253941802333, 0.004410492244584028, 0.004490610178182284, 0.0023194141776695294, 0.004550698628380977, 0.004474
586591462633, 0.0040980323035508266, 0.004386456864504551, 0.0038176195359569285, 0.003765542879118062, 0.003525189078323292, 0.003401006281245994, 0.004799064222535572, 0.004262274067427253,
0.0027560569157800284, 0.005011376746570952, 0.0032487822074093067, 0.003909755159594924, 0.004546692731701064, 0.00380560184591719, 0.0039698436097936165, 0.0038696961927957955, 0.00430633893
0906294, 0.003781566465837713, 0.004086014613511088, 0.0]
```

2. 拉普拉斯算子增强

(1). 理论背景：

- 离散情况下二阶导数：
  $\Delta^2 f=[f(x+1,y)+f(x-1,y)+f(x,y+1)+f(x,y-1)]-4f(x,y)$

  转化为拉普拉斯滤波器：

0	1	0
1	−4	1
0	1	0

改良版拉普拉斯滤波器：

−1	−1	−1
−1	8	−1
−1	−1	−1

- 将原图像和拉普拉斯滤波器卷积能够得到拉普拉斯图像，能够得到原图像更加清晰的细节信息，然后将拉普拉斯图像和原图像叠加能得到锐化的新的图像

$$g(x,y)=\begin{cases} f(x,y)-\nabla^2 f(x,y), & \text{当拉普拉斯滤波中心系数为负} \\ f(x,y)+\nabla^2 f(x,y), & \text{当拉普拉斯滤波中心系数为正} \end{cases}$$

(2). 代码实现

1. 得到原图像的灰度值矩阵，并且在四周填充0得到填充后的灰度值矩阵`padded_matrix`，避免原图像和拉普拉斯滤波器卷积时得到的拉普拉斯图像的大小和原图像不同

```
    gray_matrix=np.array(image)
    #2.填充一圈0
    padded_matrix=np.pad(gray_matrix,((1,1),(1,1)),mode='constant')
```

2. 对`padded_matrix`除去外圈0的每个像素都和滤波器做卷积操作

```
def Laplacian_Image(padded_matrix):
    #输入：零填充后的原图像矩阵
    #输出：卷积后的拉普拉斯矩阵
    #1. 卷积核
    kernel = np.array([[-1, -1, -1],
                       [-1, 8, -1],
                       [-1, -1, -1]])
    width=padded_matrix.shape[1]

    height=padded_matrix.shape[0]
    Laplacian_matrix=np.zeros((height-1,width-1))
    #2.卷积，由于卷积核中心对称，直接对应相乘
    for x in range(1,height-1):
        for y in range(1,width-1):
            for m in range(-1,1):
                for n in range(-1,1):
                    Laplacian_matrix[x][y]+=padded_matrix[x+m]
[y+n]*kernel[m+1][n+1]
    return Laplacian_matrix
```

3. 原图像叠加拉普拉斯图像，锐化增强：注意叠加过程中可能有像素的灰度值超过最大灰度值255要做限制

```
def Laplacian_Enhancement(Laplacian_matrix,gray_matrix):
    #输入：拉普拉斯图像灰度值矩阵（Laplacian_matrix）
    #输入：原图像灰度值矩阵（gray_matrix）
    #输出：锐化增强后的灰度值矩阵
    width=gray_matrix.shape[1]
    height=gray_matrix.shape[0]
    Laplacian_enhancement_matrix=np.zeros((height,width))
   #叠加
    for x in range(0,height):
        for y in range(0,width):
            Laplacian_enhancement_matrix[x][y]=gray_matrix[x][y]+Laplacian_matrix[x][y]
    Laplacian_enhancement_matrix = np.clip(Laplacian_enhancement_matrix, 0, 255)
    return Laplacian_enhancement_matrix
```

- 将位平面整合成图像矩阵：图像像素位置的灰度值=每个位平面上对应像素位置的值*该位平面上的权重

```
#将位平面加回图像,位平面取值（0，255（1））
def  merge_bit_planes_to_image(bit_planes):
    height=bit_planes[0].shape[0]
    width=bit_planes[0].shape[1]
    new_matrix=np.zeros((height,width))

    for x in range(0,height):
        for y in range(0,width):
            for  i in range(0,8):
                if bit_planes[i][x][y]!=0:
                    new_matrix[x][y]+=(pow(2,i))

    return new_matrix
```

(3). 实验结果

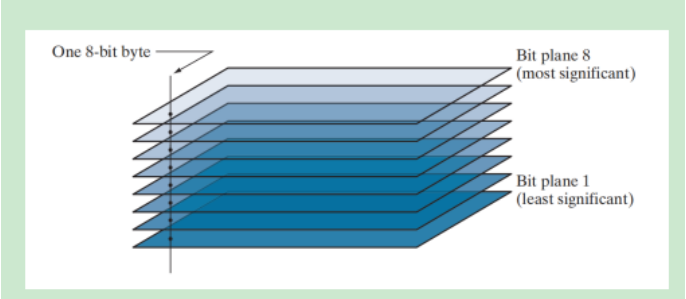
- 拉普拉斯图像提取了很多原图像的细节。
- 拉普拉斯增强后的新的图像在左半部分的细节比原图像更加清晰，但是右半部分显示出曝光的样子。

- 经过分析，原因为该部分在原图像的灰度值较高且拉普拉斯图像灰度值也高，导致叠加后超过255，为了正常显示，所有超过255的灰度值限制为255，则右半部分灰度值超过255的像素无法区分变化，全部表现为全白。

3.位平面分析：

(1).理论背景：

- 位平面：将每个像素的灰度值的每个比特位分开，不同像素的同一个比特位组成一个平面。



- 高位能表示主体信息，低位给出不同程度的细节，更改低位平面对整个图像影响小，一般可以在低位平面隐藏信息
- 位平面画图：由于位平面为二值图，取值为0时灰度值为0（全黑），取值为1时灰度值为255（全白）

(2). 代码实现：

1. 提取出位平面，都为二值图，但是为了方便后期画图，位平面的取值为0或者255
   用位操作取出灰度值的每一比特位，放在bit_planes的越前面，为越低位

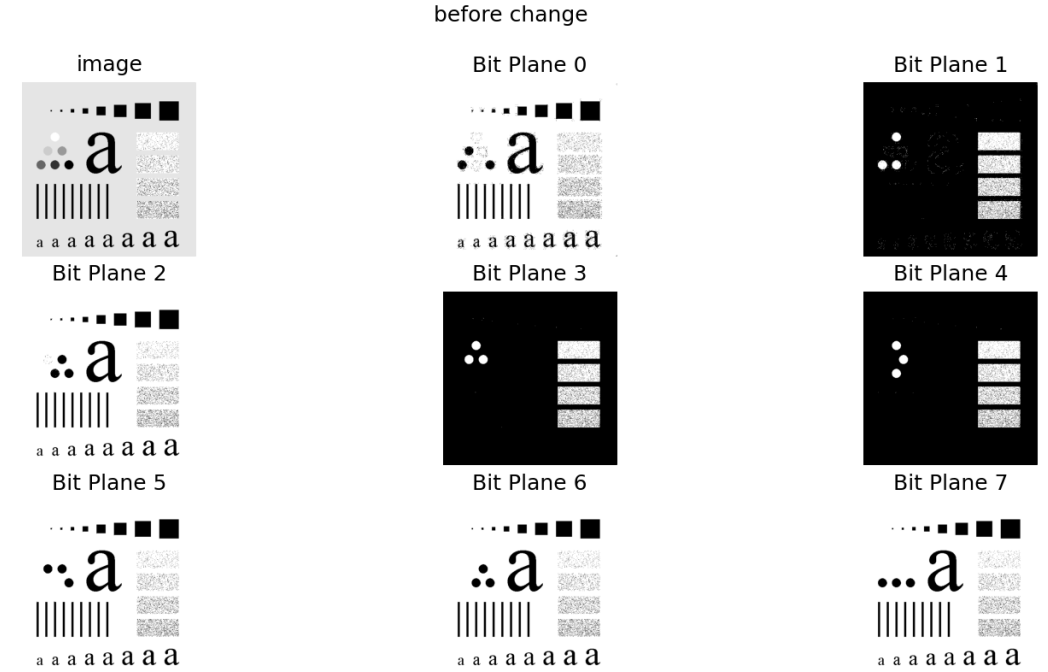
```
def Get_bit_plane(gray_matrix):
    bit_planes=[]
    for i in range(0,8):
        bit_plane=((gray_matrix>>i)&1)*255 #先提取出来的是最低位的,255是为了后期绘图得到二值图（bit=0，灰度为0，bit为1，灰度为255）
        bit_planes.append(bit_plane)
    return bit_planes
```

2. 随机改变最低位平面：将最低位平面中间条带变为全0，图上表现为最低位平面中间有一个全黑条带

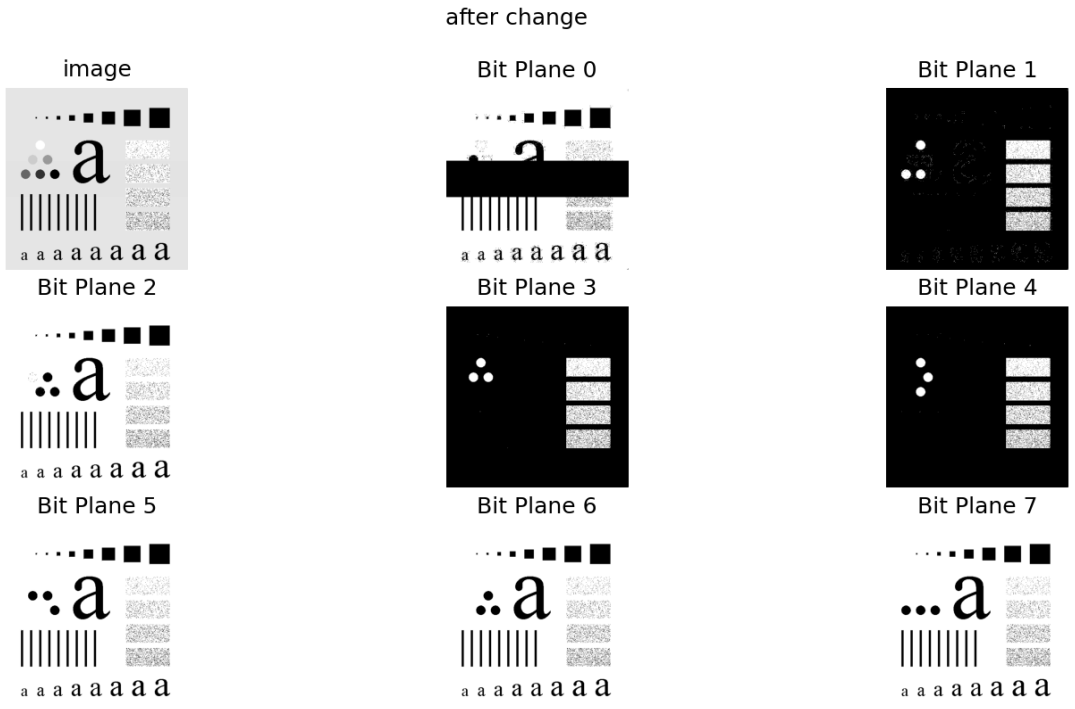
```
def change_LSB(LSB):
    height=LSB.shape[0]
    width=LSB.shape[1]
    change_height=height//2
    for i in range(-50,50):
        for j in range(0,width):
            LSB[change_height+i][j]=0  #也就是将对应bit改为0
```

(3).实验结果：

- 改变最低位平面前的原图像和位平面图像：显然最高位平面的总体图像和原图像更像，能够表示主体信息



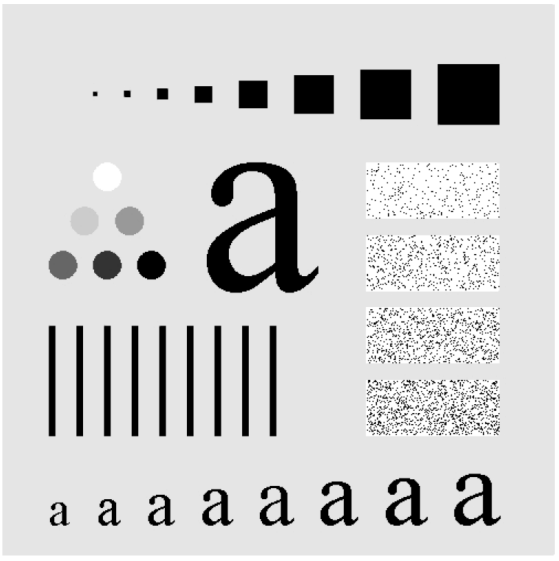
- 改变最低位平面后的新图像和位平面图像：根据更改最低为位平面的函数，显然最低位平面中间出现了一个黑色条带，其他位平面没有变化



- 对比原图像和新图像：几乎没有区别，表明改变最低位平面对图像的影响小

compare

before



after