Problem 1

Digital Image Processing

肖逸群 116033910112

Histogram Equalization

问题描述

> (a) Write a computer program for computing the histogram of an image.
> (b) Implement the histogram equalization technique.
> (c) Your program must be general to allow any gray-level image as its input.
> As a minimum, your report should include the original image, a plot of its histogram, a plot of the histogram-equalization, transformation function, the enhanced image, and a plot of its histogram.

算法思想

1. 读取原图并转化为灰度图，传入函数 `histogramProcessing` 进行计算。
2. 统计各个灰度值的出现次数 `grayCount`。
3. 求其各灰度值之前的累计出现次数，保存在字典 `grayCDF` 中，并在最后除以图像总像素数，得到灰度的概率分布累积函数，保存在字典 `grayCDF` 中，其中字典的 key 表示原本的灰度值。
4. 对累积分布函数进行均衡化，转化函数 T 如下：

$$T_{value} = round(\frac{grayCDF_i - minGray}{size - minGray} \times 255)$$

   其中，i 表示原本图像的灰度值，minGray 表示原图像中最小的灰度值，经过T转化得到一个均衡化后的灰度值，并使其修改原灰度值。
5. 对原图像的每个灰度都进行一次 T 转换，便可以得到均衡化后的图像。

程序源码

- 读取图像，并转化成灰度图，这样可以直接省去灰度图的判断。然后展示原图像和调用函数 `histogramProcessing` 均衡化后的图像

```
1   originalImage = np.array(Image.open('./resource/Fig1.jpg').convert('L'))
2
3   plt.subplot(1, 2, 1)
4   plt.imshow(originalImage, cmap = plt.get_cmap('gray'))
5   plt.title('Original Gray Scale Map')
6
7   plt.subplot(1, 2, 2)
8   enhanceImage, histogramEqualizationImage, grayCDFImage = histogramEqualizationAl
  gorithm.histogramProcessing(originalImage)
9   plt.imshow(enhanceImage, cmap = plt.get_cmap('gray'))
10  plt.title('Enhance Gray Scale Map')
11
12  plt.show()
```

以下是函数 `histogramProcessing` 处理模块：

- 读取图像信息，统计各灰度值出现次数

```
m, n =  originalImage.shape
  size = m * n
  sum = 0
  image = originalImage.copy()
  grayCount = np.zeros(256, dtype = int)
  histogramEqualization = {}

  for i in range(m):
      for j in range(n):
          grayCount[originalImage[i][j]] += 1
```

- 计算各灰度值累积出现次数，以字典型保存

```
grayCDF = dict((i, grayCount[i]) for i in range(256) if grayCount[i] != 0)

  for key in range(256):
      if key in grayCDF.keys():
          grayCDF[key] += sum
          sum = grayCDF[key]

  minGray = min(grayCDF.values())
```

- 将各像素累积出现次数除以总像素点数，即图片规模，得到概率分布累积函数，并进行均衡化操作，得到新的灰度值

```
for key in grayCDF.keys():
      histogramEqualization[key] = grayCDF[key] * 1.0 / size
      grayCDF[key] = transformFunction(grayCDF[key], minGray, size)
```

- 转化函数的定义

```
def transformFunction(pixelCDF, minGray, size):
  return round(1.0 * (pixelCDF - minGray) / (size - minGray) * 255)
```

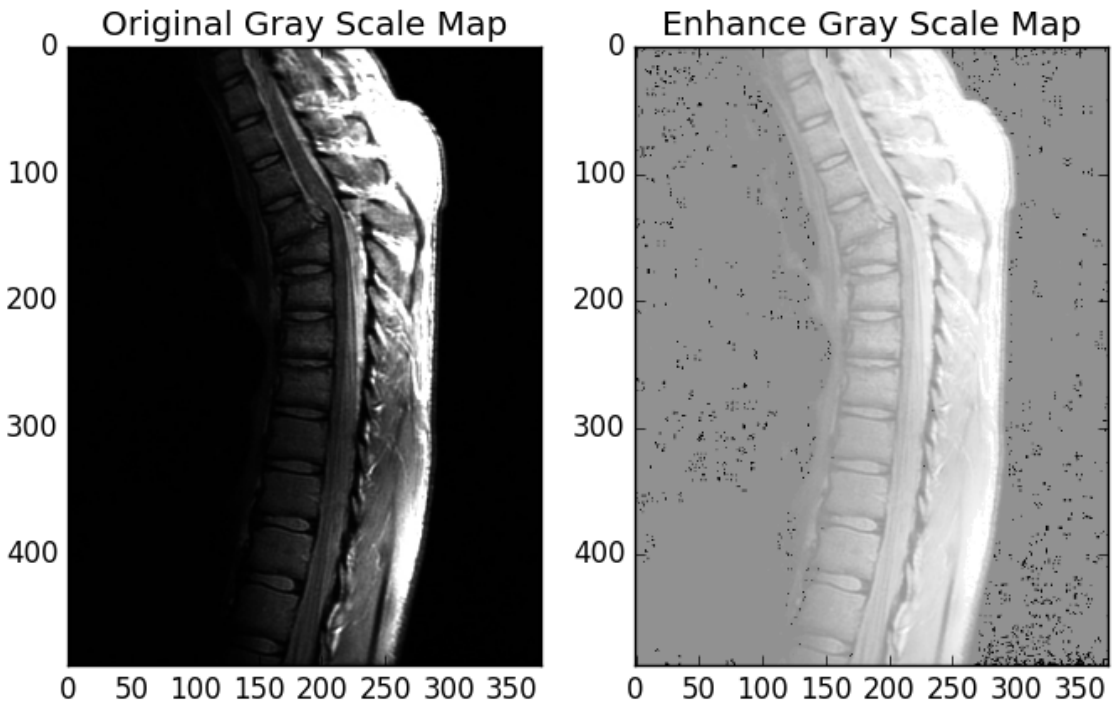
- 显示原图像灰度分布柱状图，增强图像灰度分布柱状图，原图像的概率累积分布图，转换函数图

```
plt.subplot(2, 2, 1)
  plt.hist(originalImage.flatten(), 256)
  plt.title('Original Histogram')
  plt.ylabel('Histogram Count')

  plt.subplot(2, 2, 2)
  plt.hist(enhanceImage.flatten(), 256)
  plt.title('Enhance Histogram')
  plt.ylabel('Histogram Count')

  plt.subplot(2, 2, 3)
  plt.plot(histogramEqualizationImage.keys(), histogramEqualizationImage.values())
  plt.title('Histogram Equalization')
  plt.xlabel('Gray Value')
  plt.ylabel('Relative Frequency')

  plt.subplot(2, 2, 4)
  plt.plot(grayCDFImage.keys(), grayCDFImage.values())
  plt.title('Transform Function')
  plt.xlabel('Original Gray Value')
```

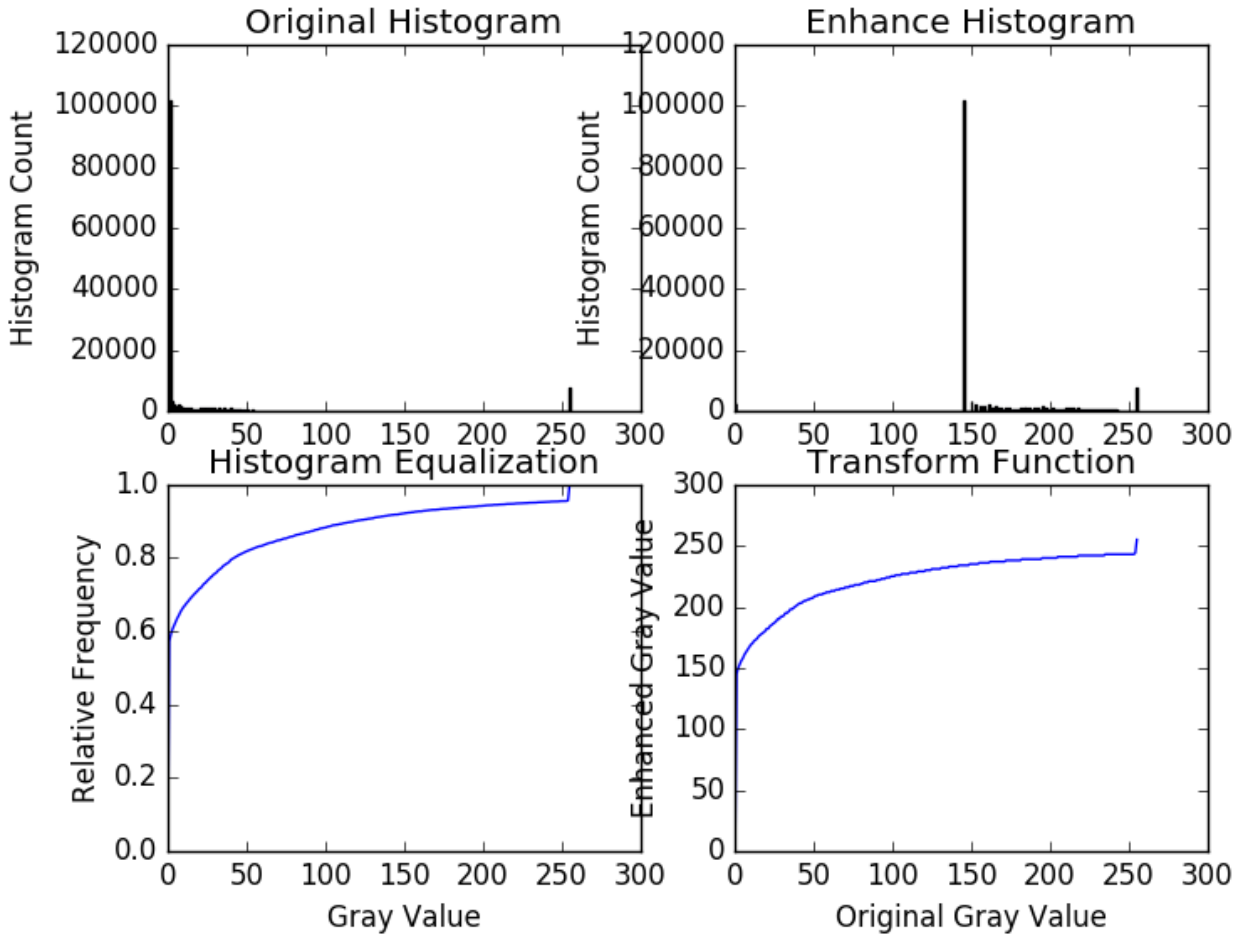
```
21  plt.ylabel('Enhanced Gray Value')
22
23  plt.show()
```

实验结果

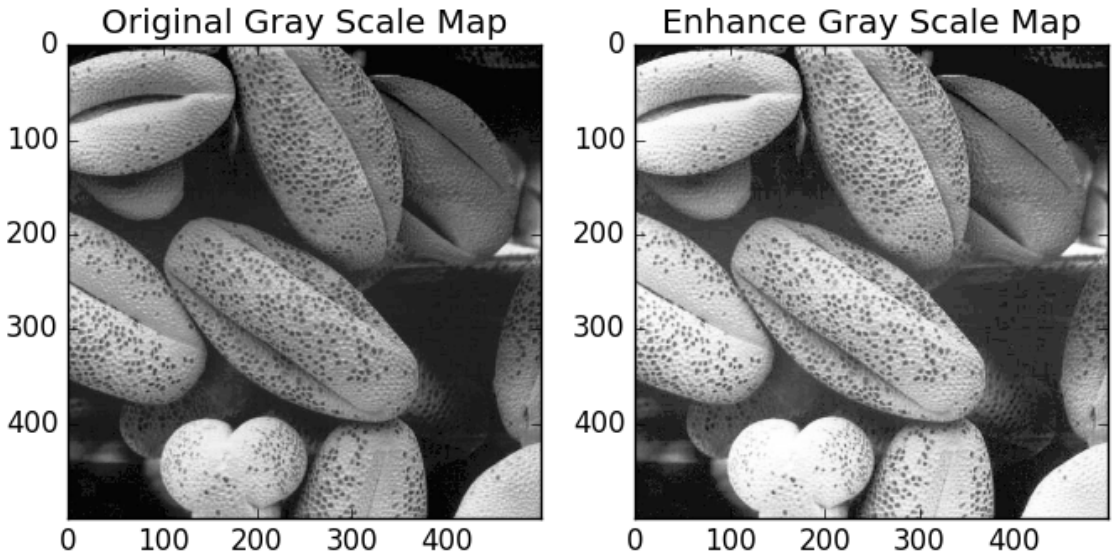
- Fig1的原始图像与增强图像



- 可以看到增强图像的灰度值是从150左右开始的，这是由于原图像中灰度为0的像素占了很大的比例，导致映射后最小的灰度值偏大



- Fig2的原始图像与增强图像



- 因为原始图像的分布不会像图1一样集中在一个灰度值上，故而效果较好