

**数字图像处理实验报告**

实验 一

题 目 数字图像处理实验一

学 院 计算学部

专 业 计算机科学与技术

学 号 22B903037

学 生 刘建

任 课 教 师 姚鸿勋 刘绍辉

哈尔滨工业大学计算机科学与技术学院

2022年秋季

实验一 报告

1. 实验内容（contents）

1. 熟悉BMP图像的结构，编程实现BMP图像的阅读和显示（不能调

用现有的库）

2. 统计图像直方图

3. 编程实现彩色BMP图像的三个分量R、G、B的显示，YIQ，HSI，YCbCr和XYZ的显示

4. 实现图像按照指定块划分，并置乱块的位置显示图像，然后尝

试恢复置乱后的图像

1. 尝试实现PSNR和SSIM两种图像评价指标

• PSNR-峰值信噪比

• SSIM-结构相似性测量指标

1. 将BMP图像转化为GIF图像

• 颜色进行量化，可以采用传统的聚类算法，也可以采用深度网络

来进行真彩色到256种颜色，以及128,64,32,16等种颜色的自动转换

• 然后再将颜色量化之后的图像保存为BMP图像，注意需要使用调色

板来保存（GIF图像可以直接调用库，与本实验的结果进行比较）

1. 实验目的（purposes）

第一个实验就是基本平台和bitmap图像的阅读，显示，和基本处理。

了解底层，熟悉平台，为后续实验做准备。

1. 实验设计、算法和流程(Design, algorithm and procedure)

针对每个具体任务实现相应的函数。

1. 任务一：读取bmp文件并显示，返回 numpy array。

**img = readinfo(file\_path)**

函数readinfo，根据位图的格式使用python的struct.unpack逐个字节进行解析。

1. 任务二：统计图像直方图。

**calHist(img,channel=0)** # 统计灰度直方图

函数calHist输入上述图像的numpy array，对指定通道的像素进行统计并绘制。

1. 任务三：编程实现彩色BMP图像的三个分量RGB的显示，YIQ，HSI，YCbCr和XYZ的显示

**rgb2rgb(img)** #输入RGB图对于的numpy array并显示

**rgb2yiq(img)** #输入img，进行RGB->YIQ坐标变换，再显示

**rgb2xyz(img)**#输入img，进行RGB->HSI坐标变换，再显示

**rgb2ycbcr(img)** #输入img，进行RGB->YCbCr坐标变换，再显示

1. 任务四：实现图像按照指定块划分，并置乱块的位置显示图像，然后尝

试恢复置乱后的图像

**PermutationFun(inputImage, blockwidth,blockheight,swap\_mode,seed)**

实现图像inputImage按照指定块(blockwidth,blockheight)划分，并按照swap\_mode置乱块的位置显示图像：先按照（blockwidth,blockheight）将图像换分为（row，col）个块，接着将图像拓展为正方形，并对图像不足的尺寸补全0，接着对（row，col）中的每个位置按照swap\_mode中的装换方式，进行seed次的位置转变，接着按照这个位置转变，对inputImage中每个（blockwidth,blockheight）块按照其位置装换后的新位置保存到相应位置的image\_swap, 返回image\_swap

**dePermutation(image,block\_width,block\_height, swap\_mode,seed)**

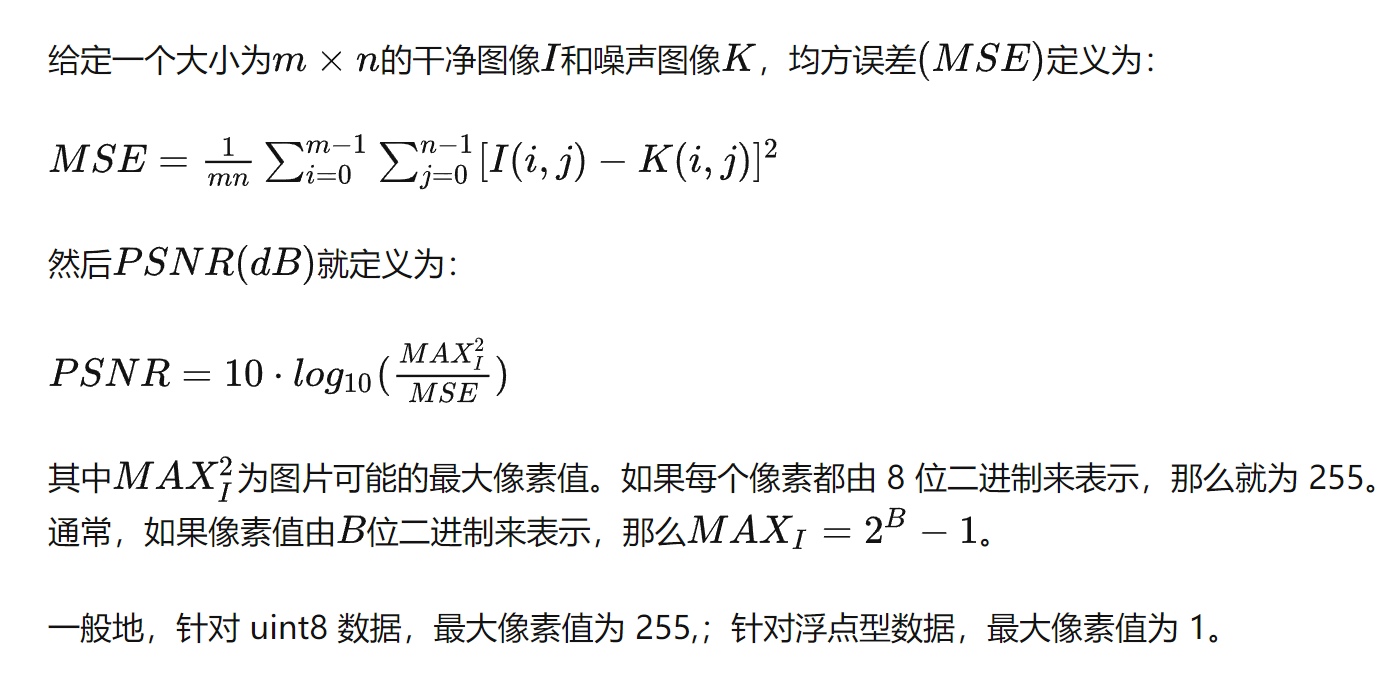
尝试恢复置乱后的图像。先按照（blockwidth,blockheight）将图像换分为（row，col）个快，接着对（row，col）中的每个位置按照swap\_mode的逆所表示的装换方式，进行seed次的位置转变，接着按照对image中每个（blockwidth,blockheight）快按照其位置装换后的新位置保存到相应位置的image\_swap, 返回image\_swap

**Permutation\_rigon(image,block\_width,block\_height, swap\_mode,rigon)**

对image指定区域reigon内的图像分块并置乱块的顺序再显示。 其中将图像image的rigon区域取出调用Permutation函数对其进行置乱，返回image\_swap，接着按照image\_swap的大小，对image图像进行拓展对齐（block\_width,block\_height），不足处补0，接着将image的region补齐（block\_width,block\_height）大小的区域设置为image\_swap的值。

1. 任务五：尝试实现PSNR和SSIM两种图像评价指标

• PSNR-峰值信噪比



函数签名：def psnr(target, ref, scale=100)

target:目标图像 ref:参考图像 scale:尺寸大小

关键代码：

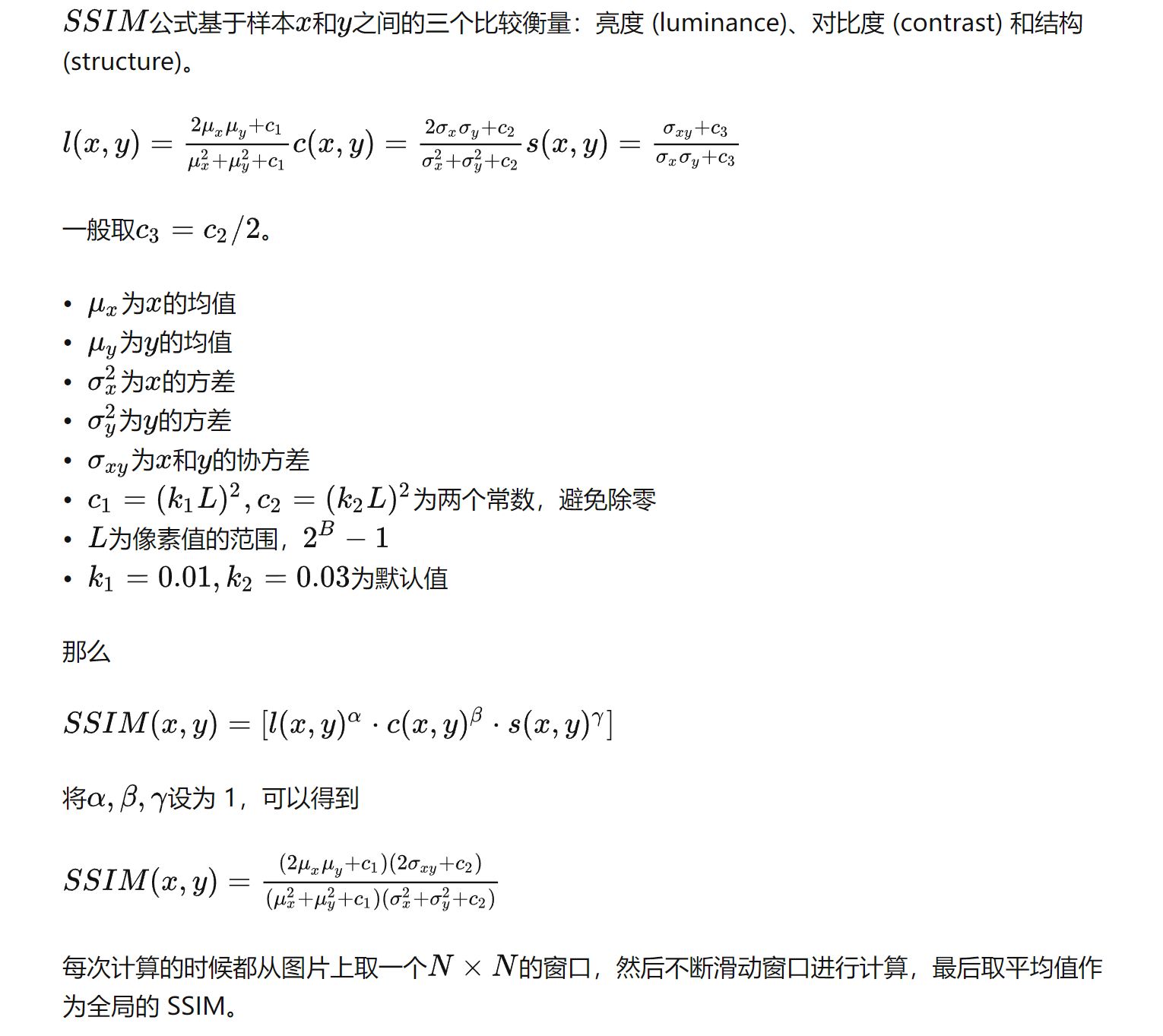
diff = ref\_data - target\_data

diff = diff.flatten('C')

rmse = math.sqrt(np.mean(diff \*\* 2.)) + 1e-5

return 10\*math.log((255\*\*2/rmse),10)

• SSIM-结构相似性测量指标



函数签名def compute\_ssim(x, y)

关键代码：

up = (2 \* ux \* uy + c1) \* (2 \* dxy + c2)

down = (ux \*\* 2 + uy \*\* 2 + c1) \* (dx + dy + c2)

return up / down

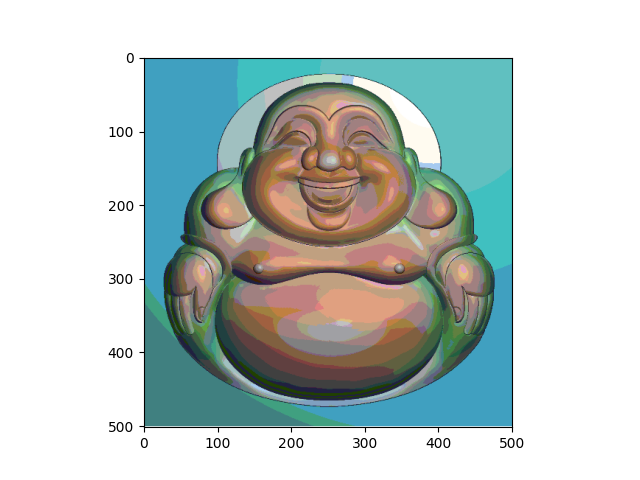
1. 任务六：将BMP图像转化为GIF图像

**BMP\_GIF(picture , k=58)**

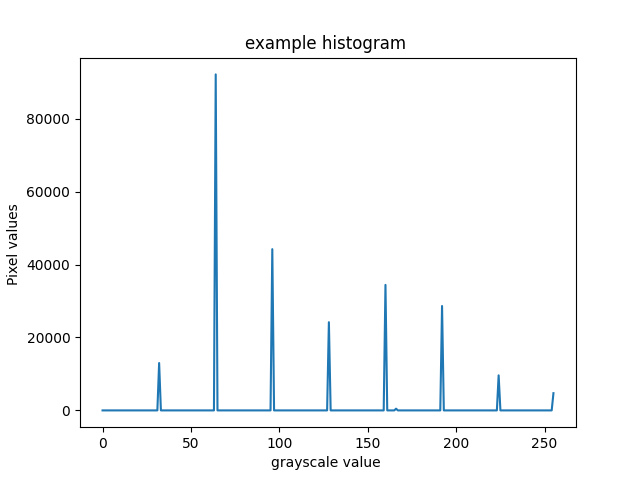
将bmp图像picture装换为颜色数量为k的GIF图像并保存为BMP文件。

首先用聚类算法将picture图像中的颜色聚类减少到k个颜色，我们构造Cluster类进行聚类，类中的kmeans方法完成聚类。Picture中每个像素点的3维向量构成一个颜色点，在其中会先初始化k个中心点为picture中k个不同的颜色点，接着遍历这个picture的每个像素点获取相应颜色点，计算与每个颜色点最近的中心点，加入相应中心点的集合，最后计算每个中心点集合的均值作为新的中心点值，不断进行遍历直到中心点不变或者达到设置的最大遍历数，返回k个中心点，和每个中心点集合。

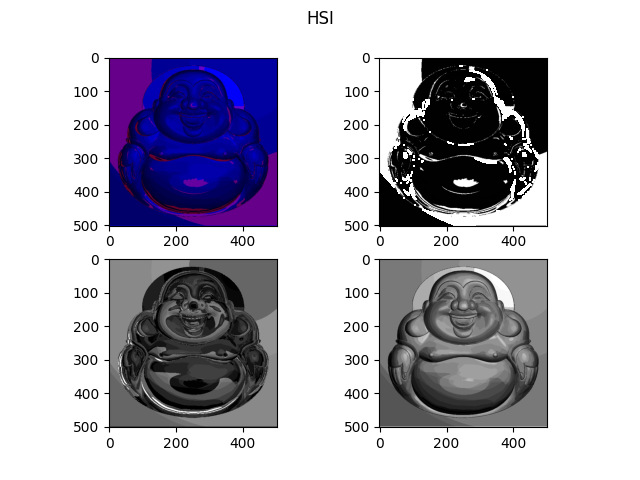
1. 实验结果(results)
2. 任务一：读取bmp文件并显示，返回 numpy array。

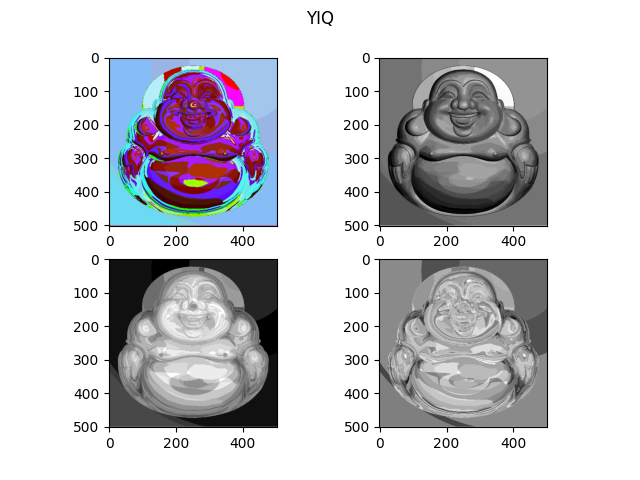


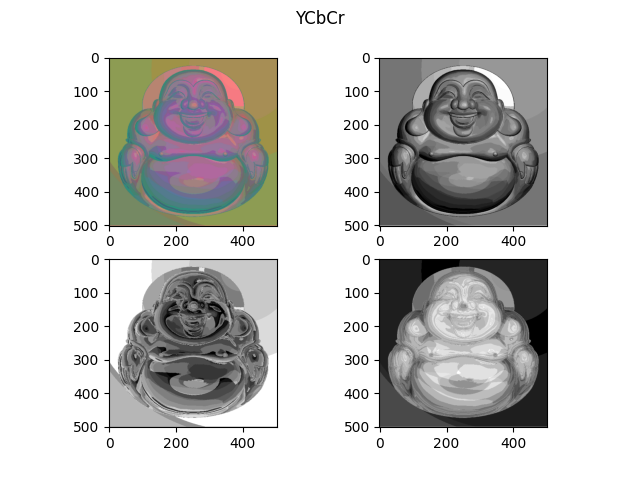
1. 任务二：统计图像直方图。

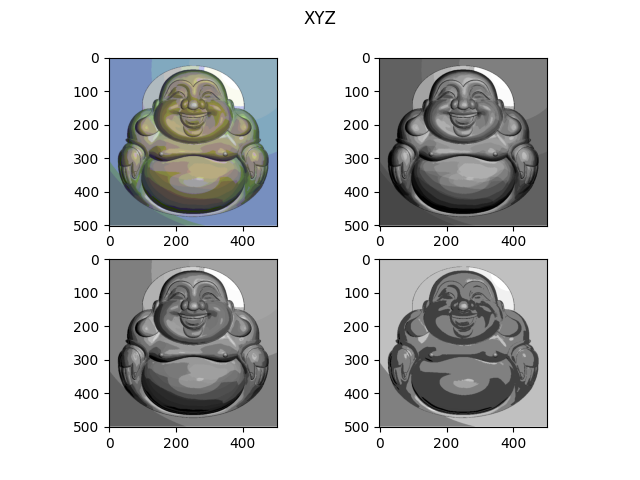


1. 任务三：编程实现彩色BMP图像的三个分量RGB的显示，YIQ，HSI，YCbCr和XYZ的显示



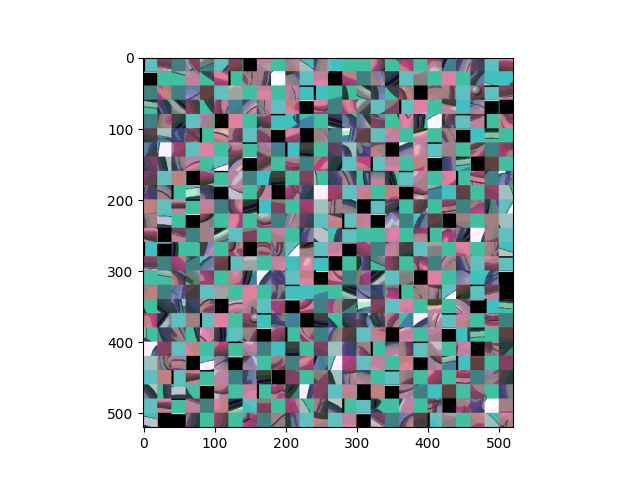


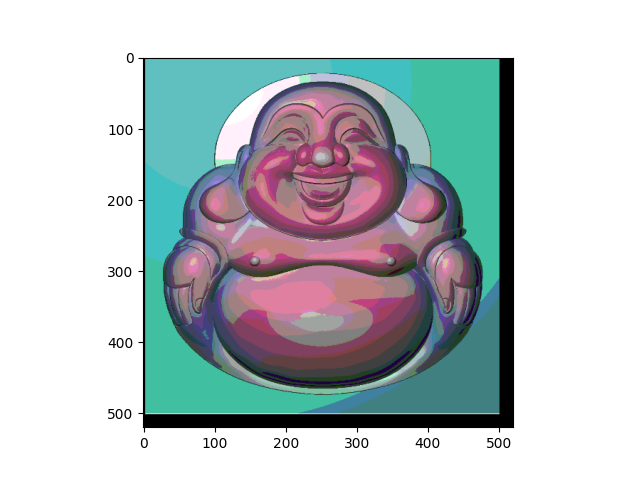




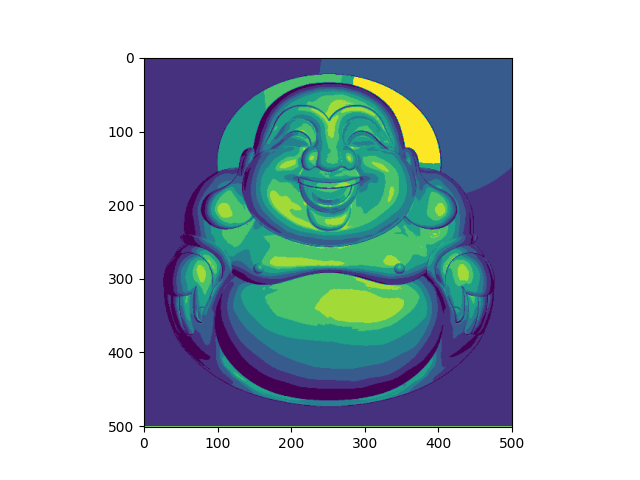
1. 任务四：实现图像按照指定块划分，并置乱块的位置显示图像，然后尝

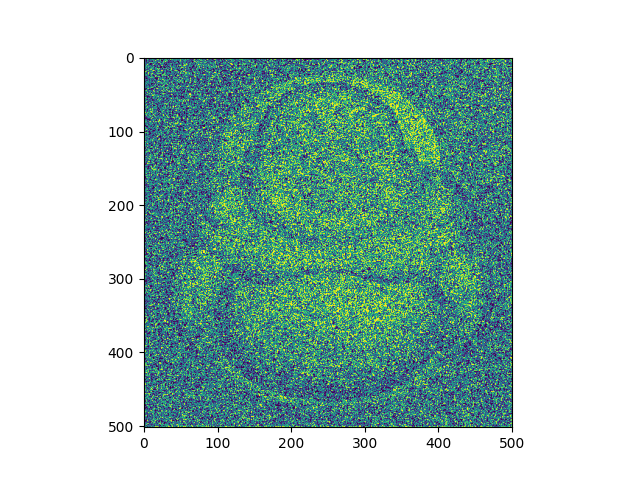
试恢复置乱后的图像





1. 任务五：尝试实现PSNR和SSIM两种图像评价指标





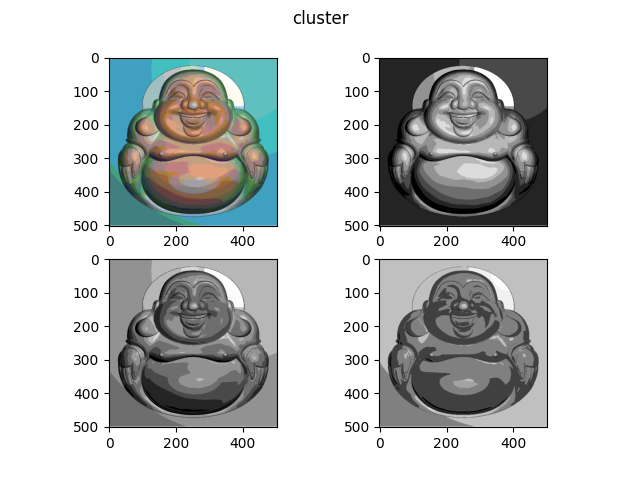
**psnr: 25.999280782541806**

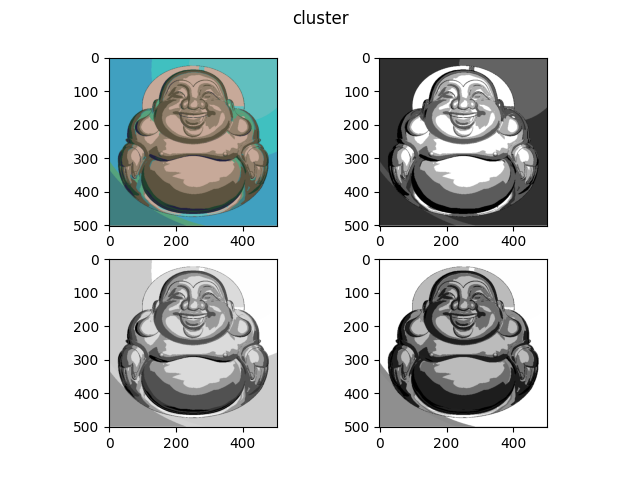
**ssim: -5.594568550728403e-06**

1. 任务六：将BMP图像转化为GIF图像

聚类：前者是58种颜色（采用的是8bit伪彩色图，该test.bmp经统计只有58种颜色）

后者是10种颜色。





1. 结论(conclusion)

本次实验设计，将bmp图读取成numpy.array后，就可以基于它做相应的运算。其中置换和聚类那块儿挺耗精力的，出现了一些易错的bug。

比如:聚类的时候，算distance，之前numpy.array指定的是np.uint8，这在开方的时候会变成0，导致后续聚类中心算错，维度也有不匹配的。

总体而言，通过这个实验，对颜色空间，和矩阵运算有了新的认识。

1. 参考文献(reference)

代码中某些模块的设计有些参考链接

***Digital Image Processing***

**Mark sheet of experimentation**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No | Student No. | Name | Score of Part 1  (code) | Score of Part 2  (content) | Score of Part 3  (result) | Score of Part 4  （report） | Total Score |
| 1 |  |  |  |  |  |  |  |

Signature: Date: