

**《数字图像处理》**

**作业3：图像量化处理**

**姓 名： 黄宇快**

**班 级： 2018211602**

**学 号： 2018213350**

**指导教师： 李学明**

北京邮电大学数字媒体与设计艺术学院

2020年 9 月 30 日

## 作业目的：

理解图像量化的基本原理，掌握均匀量化和k-means量化方法

## 作业内容：

（1）选取一张灰度图像，采用均匀量化方法将其灰度值量化为32个灰度等级，比较量化前后图像质量的差异

（2）选取一张彩色图像，采用K-means方法将图像的颜色量化为128种颜色，比较量化前后图像质量的差异

作业提示：

（1） k-means是机器学习中的经典聚类算法，scikit-learn中有相关的函数实现

（2）在K-means算法中将彩色图像每个像素看成一个(r,g,b)三个值构成的三维向量

数字图像处理

问题1：图像的采样和量化

在搜集和阅读资料后，在从采样和量化的角度，我们可以整理出均匀采样和非均匀采样，同样的也有均匀量化和非均匀量化。

采样就是把一幅图像分个成M\*N个网格，每一个网格用一个亮度值表示。一个网格为一个像素。作业2中摄像机工作原理就是对实际的景物进行采样，得到8bit深度图像。现在的相机以及可以达到1亿像素，如18年推出的哈苏新推了一款H6D-400c中画幅相机，拥有1亿像素中画幅CMOS传感器，而我们的人眼其实是一台像素高达5.76亿的“超级相机”。

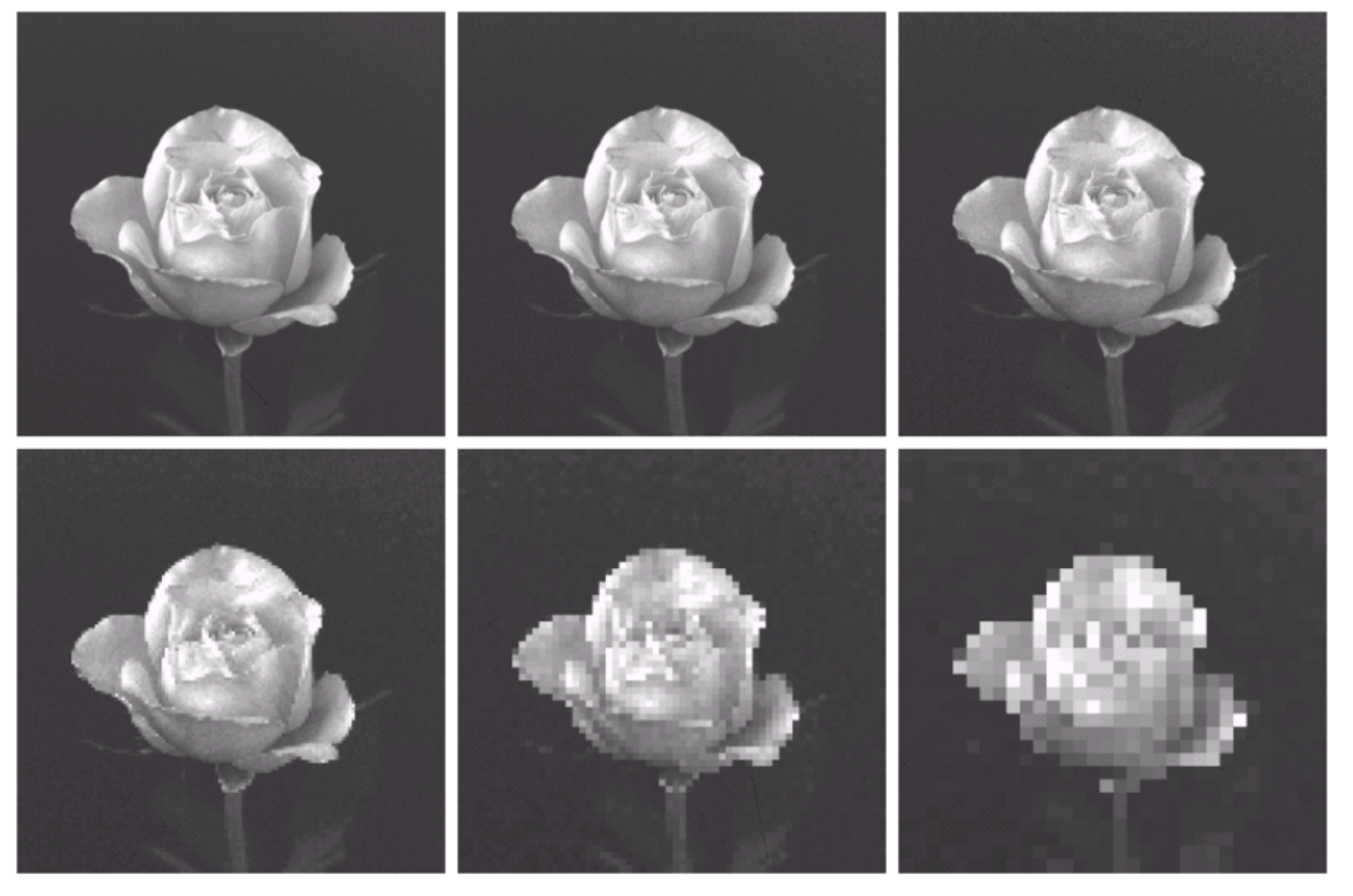


图1

在这里我们讨论一张灰度图像，采用均匀量化方法将其灰度值量化为32个灰度等级。要做到这点。

我们要划分区间，量化等级为2：使用2中灰度级来表示图片小于128取0，大于128取128。

图片包含 图形用户界面

描述已自动生成

图2 量化等级2

量化等级为4：使用4中灰度级显示图片有四个颜色区间0-64取0，64-128取64，128-192，取128，192-255取192。

图片包含 图形用户界面

描述已自动生成

图3 量化等级4

图片包含 图形用户界面

描述已自动生成

图4 量化等级8

图片包含 游戏机, 监控, 钟表

描述已自动生成

图5 量化等级16

图片包含 游戏机, 监控, 桌子, 电脑

描述已自动生成

图6 量化等级32

电子设备的屏幕

描述已自动生成

图7 量化等级64

图片包含 图形用户界面

描述已自动生成

图8 量化等级128

图片包含 图形用户界面

描述已自动生成

图9 量化等级256

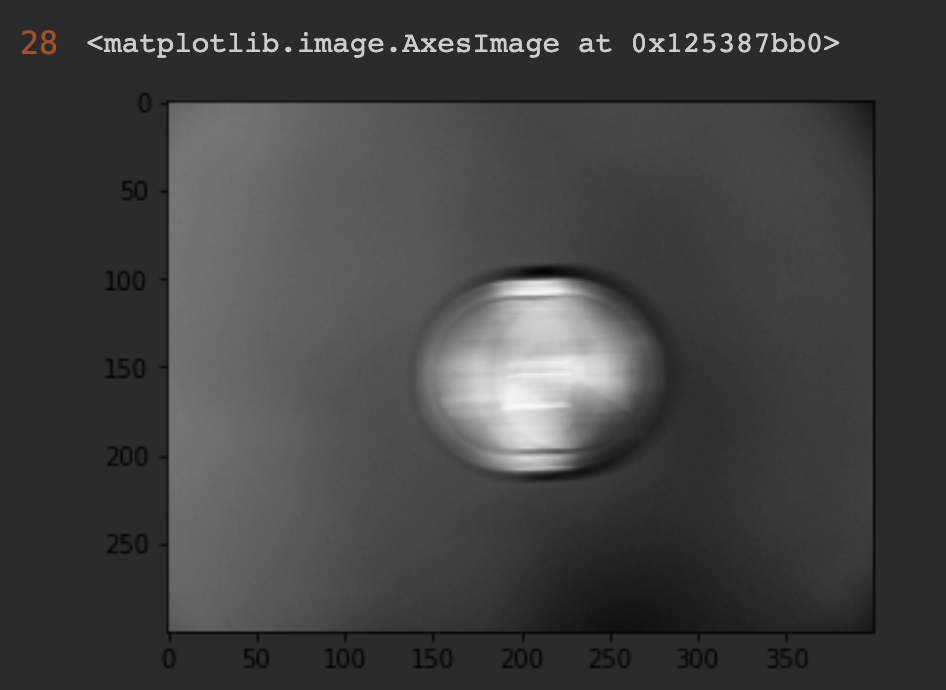


图10 原图

在这个过程通过一个算法实现判断以及分区间划分出量化等级，我们还可以看出运行时间是二倍的关系在对区间进一步一分二的过程。

算法生成的图像和原图没有区别，证明算法正确性。

算法如下：

def imgQuantification(img,grade):

tic = time.time()

compare=[i for i in range(0,256,int(256/grade))]

for n in range(grade-1):

for i in range(img.shape[0]):

for j in range(img.shape[1]):

if img[i][j]>=compare[n] and img[i][j]<compare[n+1]:

img[i][j]=compare[n]

toc = time.time()

print(toc-tic,end="s")

return img

我们在观察发现量化等级一级一级提高使得图像更贴近原图，在低量化的图像在无损压缩后，体积可以缩小很多，这与量化等级有关。

（2）选取一张彩色图像，采用K-means方法将图像的颜色量化为128种颜色，比较量化前后图像质量的差异

非均匀量化：有K-Means聚类是最常用的聚类算法，最初起源于信号处理，其目标是将数据点划分为K个类簇，找到每个簇的中心并使其度量最小化。该算法的最大优点是简单、便于理解，运算速度较快，缺点是只能应用于连续型数据，并且要在聚类前指定聚集的类簇数。

下面是K-Means聚类算法的分析流程，步骤如下：

第一步，确定K值，即将数据集聚集成K个类簇或小组。

第二步，从数据集中随机选择K个数据点作为质心（Centroid）或数据中心。

第三步，分别计算每个点到每个质心之间的距离，并将每个点划分到离最近质心的小组，跟定了那个质心。

第四步，当每个质心都聚集了一些点后，重新定义算法选出新的质心。

第五步，比较新的质心和老的质心，如果新质心和老质心之间的距离小于某一个阈值，则表示重新计算的质心位置变化不大，收敛稳定，则认为聚类已经达到了期望的结果，算法终止。

第六步，如果新的质心和老的质心变化很大，即距离大于阈值，则继续迭代执行第三步到第五步，直到算法终止。

主要代码

#对像素空间使用k-means聚类，将1600万种颜色（255x255x255=16581375）缩减到128种颜色。

#因为我们处理的是非常大的数据集，所以将使用MiniBatchKMeans算法，这种算法速度比k-means速度快

from sklearn.cluster import MiniBatchKMeans

kmeans = MiniBatchKMeans(128)

kmeans.fit(data)

nonUniformImgQuantification = kmeans.cluster\_centers\_[kmeans.predict(data)]

水中的建筑

描述已自动生成

对比原图与128量化等级的非均匀量化，我发现在宝塔的颜色部分有点暗淡，明显看出没有原图的颜色那么丰富，其次从天空的过度颜色来看，可以看出天空的一些边界颜色过度很不好，如果是在观察图八的均匀量化来看，非均匀量化表现的一点也不好。但是我们不能单单从图片中不足的部分来评判k-means算法的成功与否，从主要建筑物来看，k-means表现是要比均匀量化要好的，还原度上也比均匀量化高。所以我们要根据不同类型的图片来选择相应的算法，这样才是图像处理的最优解，每个算法有每个算法优秀的部分。

代码：

from sklearn.datasets import load\_sample\_image

china = load\_sample\_image("china.jpg")

#将像素数据转换成三维颜色空间中的一群数据点

data = china/256

data = data.reshape(427\*640, 3)

#对像素空间使用k-means聚类，将1600万种颜色（255x255x255=16581375）缩减到16种颜色。

#因为我们处理的是非常大的数据集，所以将使用MiniBatchKMeans算法，这种算法速度比k-means速度快

from sklearn.cluster import MiniBatchKMeans

kmeans = MiniBatchKMeans(128)

kmeans.fit(data)

nonUniformImgQuantification = kmeans.cluster\_centers\_[kmeans.predict(data)]

#%%

#用计算结果对原始像素重新着色，即每个像素被指定为距离其距离最近的簇中心点的颜色。

chinaQuantification= nonUniformImgQuantification.reshape(china.shape)

fig, ax = plt.subplots(1, 2, figsize=(16, 6),

subplot\_kw=dict(xticks=[], yticks=[]))

fig.subplots\_adjust(wspace=0.05)

ax[0].imshow(china)

ax[0].set\_title("Original Image")

ax[1].imshow(chinaQuantification)

ax[1].set\_title('128-color Image')