人工智能实验——基于 k -means 聚类算法的图片色彩分割

一、实验内容

通过 k -means 算法，对图片中相近的颜色进行聚类，并对同一簇用同一种颜色代替。以此，可以将原图片划分成不同的区域，并且可以对原图片进行压缩。

二、实验要求

实验要求：

1.对于聚类簇的数目，自己选取不少于 3 种情况，且这 3 种情况要合理。

2. k -means 算法要求自己实现。 要求代码能得到聚类之后的图像，主代码放在脚本 KmSegmentation.py 中。代码要简洁工整，尽量使用向量化编程，必要的地方要有注释。

3. 实验报告中要包括原图片在不同簇下，进行分割得到的图像，并有簇的数目的选取对图像分割的影响的分析

三、实验过程

1．读取图片

```
im = Image.open('Sea.jpg')
print im.mode,im.size,im.format
#RGB  8-bit 0~255
pix = im.load()
width = im.size[0]  #481
height = im.size[1] #321
```

2．Kmeans 聚类

K-Means 的算法如下：

1. 随机在图中取 K 个中心点。

2. 然后对图中的所有点求到这 K 个中心点的距离，假如点 Pi 离中心点 Si 最近，那么 Pi 属于 Si 点群。

3. 接下来，重新计算 si 点群的中心点。

4. 然后重复第 2）和第 3）步，直到，中心点没有移动，算法结束

```
while flag==1:
    next_k_class = [[0 for i in range(3)] for i in range(k)]
    k_class_num = [0 for i in range(k)]
    for x in range(width):
        for y in range(height):
            r, g, b = pix[x, y]
            dist = -1
            for i in range(k):#计算每个点离那个中心点最近
                tmp1 = (r - k_class[i][0]) * (r - k_class[i][0])
                tmp2 = (g - k_class[i][1]) * (g - k_class[i][1])
                tmp3 = (b - k_class[i][2]) * (b - k_class[i][2])
                if dist == -1:
                    pix_class[x][y] = i
                    dist = tmp1 + tmp2 + tmp3
                elif tmp1 + tmp2 + tmp3 < dist:
                    pix_class[x][y] = i
                    dist = tmp1 + tmp2 + tmp3
            next_k_class[pix_class[x][y]][0] += r
            next_k_class[pix_class[x][y]][1] += g
            next_k_class[pix_class[x][y]][2] += b
            k_class_num[pix_class[x][y]] += 1
    for x in range(k):
        for y in range(3):
            if k_class_num[x] != 0:
                next_k_class[x][y] = next_k_class[x][y] / k_class_num[x]#重新计算中
心点，这里取平均值
    print next_k_class
    flag=0
    for x in range(k):
        if next_k_class[x][0:3] != k_class[x][0:3]: #还有类未固定
            flag = 1
            break
    for x in range(k):
        k_class[x][0:3] = next_k_class[x][0:3]
#聚类结束
```

3．显示图片

```
im_new=Image.new("RGB",(481,321))
for x in range(width):
    for y in range(height):
        r,g,b= k_class[pix_class[x][y]][0:3]
```

```
        im_new.putpixel((x,y),(r,g,b))
im_new.show()
im_new.save('sea2.jpg')
```

四、实验结果

K=4:

K=8:

K=16:

K=32:

从中，可以明显的看出，随着 k 的增大，聚类之后的图片越来越接近于原始的图片。但算法的用时也会随着 k 的增大明显变长，压缩效果随 k 的增大变差。所以，对该算法，需要合理的选取 k 的大小。此外，k 个初始中心点的选取对图片色彩的分割有着重要的影响。