数字图像处理第一次作业实验报告

范睿霖 自动化66班 2160504145

提交日期： 2019.3.5

摘要：

简单介绍BMP格式的图像文件，将lena图片以灰度8-1逐级递减显示，计算lena图片的均方差，将lena图片分别用近邻，双线性，双三次插值法zoom到2048\*2048；把lena和elain图像分别水平shear和旋转30度

BMP格式介绍：

BMP（全称Bitmap）是Windows操作系统中的标准图像文件格式，可以分成两类：设备有向量相关[位图](https://baike.baidu.com/item/%E4%BD%8D%E5%9B%BE/1017781" \t "_blank)（DDB）和设备无向量相关位图（DIB），使用非常广。它采用位映射存储格式，除了图像深度可选以外，不采用其他任何压缩，因此，BMP文件所占用的空间很大。BMP文件的图像深度可选lbit、4bit、8bit及24bit。BMP文件存储数据时，图像的扫描方式是按从左到右、从下到上的顺序。由于BMP文件格式是Windows环境中交换与图有关的数据的一种标准，因此在Windows环境中运行的图形图像软件都支持BMP图像格式。

Lena图像灰度逐级递减显示：

8级灰度显示：



7级灰度显示：



6级灰度显示：



5级灰度显示：



4级灰度显示：



3级灰度显示：



2级灰度显示：



1级灰度显示：



根据公式计算，lena图像的均值为98.660095,方差为2783.965115。

使用近邻法zoom图像：



使用双线性：



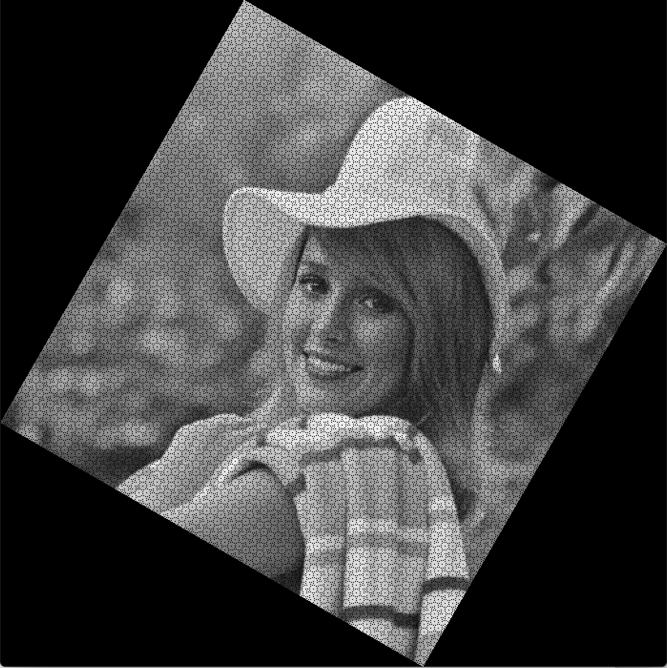
使用双三次插值法：



设置参数为0.5，进行水平shear：

设置参数为30度，进行旋转：

代码：

from PIL import Image  
  
img = Image.open('lena.BMP')  
grey = img.convert('L')  
flag = 1  
n=int(input('insert n:'))  
  
if n==1:  
 for i in range(0, 511):  
 for j in range(0, 511):  
 if grey.getpixel((i,j)) > 128:  
 grey.putpixel((i, j), 255)  
 else:  
 grey.putpixel((i, j), 0)  
elif n==2:  
 for i in range(0, 511):  
 for j in range(0, 511):  
 t=grey.getpixel((i, j))  
 t=int(t/64)  
 grey.putpixel((i,j),t\*64)  
elif n==3:  
 for i in range(0,511):  
 for j in range (0,511):  
 t=grey.getpixel((i,j))  
 t=int(t/32)  
 grey.putpixel((i,j),t\*32)  
elif n==4:  
 for i in range(0,511):  
 for j in range (0,511):  
 t=grey.getpixel((i,j))  
 t=int(t/16)  
 grey.putpixel((i,j),t\*16)  
elif n==5:  
 for i in range(0,511):  
 for j in range (0,511):  
 t=grey.getpixel((i,j))  
 t=int(t/8)  
 grey.putpixel((i,j),t\*8)  
elif n==6:  
 for i in range(0,511):  
 for j in range (0,511):  
 t=grey.getpixel((i,j))  
 t=int(t/4)  
 grey.putpixel((i,j),t\*4)  
elif n==7:  
 for i in range(0,511):  
 for j in range (0,511):  
 t=grey.getpixel((i,j))  
 t=int(t/2)  
 grey.putpixel((i,j),t\*2)  
elif n==8:  
 pass  
  
else :  
 flag = 0  
  
if flag==1:  
 grey.show()  
else:  
 print('incorret number')

from PIL import Image  
  
img = Image.open('lena.BMP')  
grey = img.convert('L')  
n=0;v=0  
for i in range(0,511):  
 for j in range (0,511):  
 n=n+grey.getpixel((i,j))  
n=n/512\*\*2  
for i in range(0,511):  
 for j in range (0,511):  
 v=v+(grey.getpixel((i,j))-n)\*\*2  
v=v/512\*\*2  
  
print('n=%f,v=%f'%(n,v))

from PIL import Image  
#from shear import \*  
import math  
import numpy as np  
img=Image.open('lena.BMP')  
grey=np.matrix(img.convert('L'))  
n=int(input('select method'))  
x\_1 = np.zeros((2048, 2048), dtype=int)  
x\_2 = np.zeros((2048, 2048), dtype=int)  
if n==1:  
  
 for i in range (2048):  
 for j in range (2048):  
 x\_1[i,j]=grey[int(i/4),int(j/4)]  
 img\_1=Image.fromarray(np.uint8(x\_1))  
 img\_1.show()  
elif n==2:  
 for i in range(2048):  
 for j in range(2048):  
 x, y = i / 4, j / 4  
 xx, yy = int(x), int(y)  
 a, b = x - xx, y-yy  
 para = np.matrix([[(1-a)\*(1-b), a\*(1-b)], [(1-a)\*b, a\*b]])  
 if xx < 510 and yy < 510:  
 x\_2[i, j] = np.sum(np.multiply(para, grey[xx:xx+2, yy:yy+2]))  
 #x\_2[i,j]=math.floor(grey[int(i/4),int(j/4)]\*(4-i%4)\*(4-j%4)/16+grey[int(i/4)+1,int(j/4)]\*(4-i%4)\*(j%4)/16+grey[int(i/4),int(j/4)+1]\*(i%4)\*(4-j%4)/16+grey[int(i/4)+1,int(j/4)+1]\*(i%4)\*(j%4)/16)  
 img\_2=Image.fromarray(np.uint8(x\_2))  
 img\_2.show()  
elif n==3:  
 img\_3=img.convert('L').resize((2048,2048),Image.BICUBIC)  
 img\_3.show()

from PIL import Image  
import numpy as np  
import math  
  
  
img\_1=Image.open ('lena.BMP')  
img\_2=Image.open ('elain.BMP')  
grey\_1=np.matrix(img\_1.convert('L'))  
grey\_2=np.matrix(img\_2.convert('L'))  
n = int(input('method select:'))  
if n == 1:  
 shear\_1=img\_1.transform((512,770),Image.AFFINE,(1,0,0,0.5,1,-250))  
 shear\_2=img\_2.transform((512,770),Image.AFFINE,(1,0,0,0.5,1,-250))  
 shear\_1.show()  
 shear\_2.show()  
elif n==2:  
   
 theta = 30  
 theta = theta/180\*math.pi  
 x\_1=np.zeros((2000,1000),dtype=int)  
 x\_2=np.zeros((2000,1000),dtype=int)  
 t=np.matrix([[math.cos(theta),math.sin(theta)],[-1\*math.sin(theta),math.cos(theta)]])  
 for i in range (np.shape(grey\_1)[0]):  
 for j in range (np.shape(grey\_1)[1]):  
 co = t\*np.matrix([i+1000,j]).T  
 xx = math.floor(co[0])  
 yy = math.floor(co[1])  
 #img.putpixel((int(0.866\*i-0.5\*j)+250,int(0.866\*i+0.5\*j)),grey.getpixel((i,j)))  
 #if xx < np.shape(grey)[0] and yy < np.shape(grey)[1]:  
 x\_1[xx,yy]=grey\_1[i,j]  
 x\_2[xx,yy]=grey\_2[i,j]  
 for i in range(np.shape(x\_1)[0]):  
 if np.max(x\_1[i,:])>0:  
 x\_1 = x\_1[i:-1,:]  
 break  
 for i in range(-1,-1\*np.shape(x\_1)[0],-1):  
 if np.max(x\_1[i,:])>0:  
 x\_1=x\_1[0:i+1,:]  
 break  
 for i in range (np.shape(x\_1)[1]):  
 if np.max(x\_1[:,i])>0:  
 x\_1 = x\_1[:,i:]  
 break  
 for i in range (-1,-1\*np.shape(x\_1)[1],-1):  
 if np.max(x\_1[:,i])>0:  
 x\_1 = x\_1[:,0:i+1]  
 break  
 for i in range(np.shape(x\_2)[0]):  
 if np.max(x\_2[i,:])>0:  
 x\_2 = x\_2[i:-1,:]  
 break  
 for i in range(-1,-1\*np.shape(x\_2)[0],-1):  
 if np.max(x\_2[i,:])>0:  
 x\_2=x\_2[0:i+1,:]  
 break  
 for i in range (np.shape(x\_2)[1]):  
 if np.max(x\_2[:,i])>0:  
 x\_2 = x\_2[:,i:]  
 break  
 for i in range (-1,-1\*np.shape(x\_2)[1],-1):  
 if np.max(x\_2[:,i])>0:  
 x\_2 = x\_2[:,0:i+1]  
 break  
 img\_1 = Image.fromarray(np.uint8(x\_1))  
 img\_2 = Image.fromarray(np.uint8(x\_2))  
 img\_1.show()  
 img\_2.show()