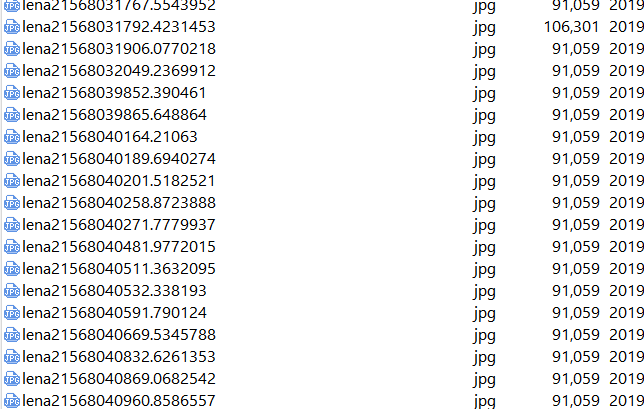
图像处理作业-刘继东

# 第一周作业：打开图像，显示图像，存储图像，对一张图片进行缩放，观察其分辨率，降低灰度处理

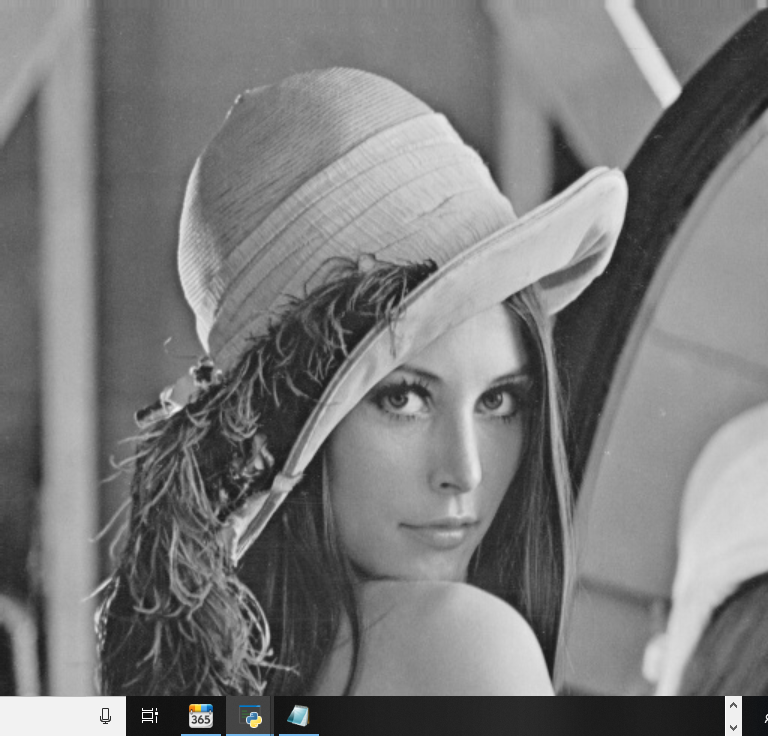
显示图片：



存储：增加时间戳作文文件名，保存如下：



放大1.5倍：



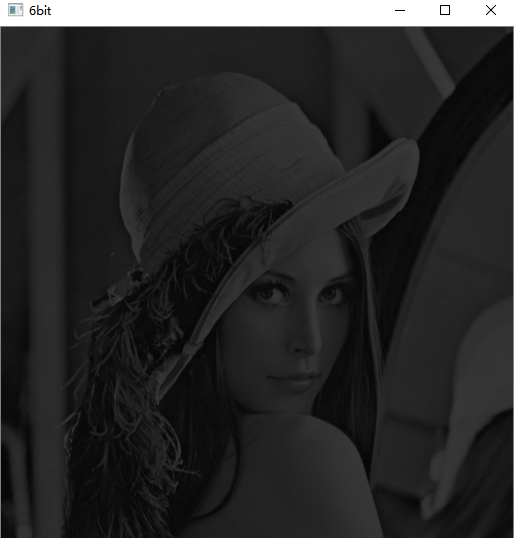
缩小0.5倍：



降低灰度值（每个像素灰度值-100）：



降低灰度级，6bit



源码：

###2019年9月9日 第一周作业：打开图像，显示图像，存储图像，对一张图片进行缩放，观察其分辨率，降低灰度处理  
import cv2 as cv  
import time  
import numpy as np  
  
img = cv.imread("./source/lena.jpg",cv.IMREAD\_GRAYSCALE) #读入做灰色处理  
print(img.shape) #打印大小  
(height,width) = img.shape  
  
cv.imshow("lena",img) #显示  
rename = str(time.time())  
print(rename,height,width)  
cv.imwrite("./dest/lena2"+rename+".jpg",img) #存错  
  
#放大一倍  
largedim = (int(height\*1.5),int(width\*1.5))  
img\_enlarge=cv.resize(img,largedim) #1data 2 (高，宽)  
cv.imshow("large",img\_enlarge);  
print(img\_enlarge.shape)  
  
###缩小一半,方法一  
img\_enMin=cv.resize(img,None,0.5,0.5,cv.INTER\_AREA) #1data 2 (高，宽)  
cv.imshow("minor1",img\_enMin);  
print(img\_enMin.shape)  
  
###缩小一半,方法二  
mindim = (int(height\*0.5),int(width\*0.5))  
img\_enMindim=cv.resize(img,mindim,cv.INTER\_AREA) #1data 2 (高，宽)  
cv.imshow("minor2",img\_enMindim);  
print(img\_enMindim.shape)  
  
  
##@改变每个像素的灰度值，每个像素降低100, 生成同大小的二维数组  
subval = np.full(img.shape,100,dtype=np.uint8)  
subimg = cv.subtract(img,subval)  
cv.imshow("subimg",subimg)

###降低灰度级数8bit图像降为6bit。  
destImg = img//4  
cv.imshow("6bit",destImg )  
print(destImg)  
  
cv.waitKey(0)

# 第二周作业：

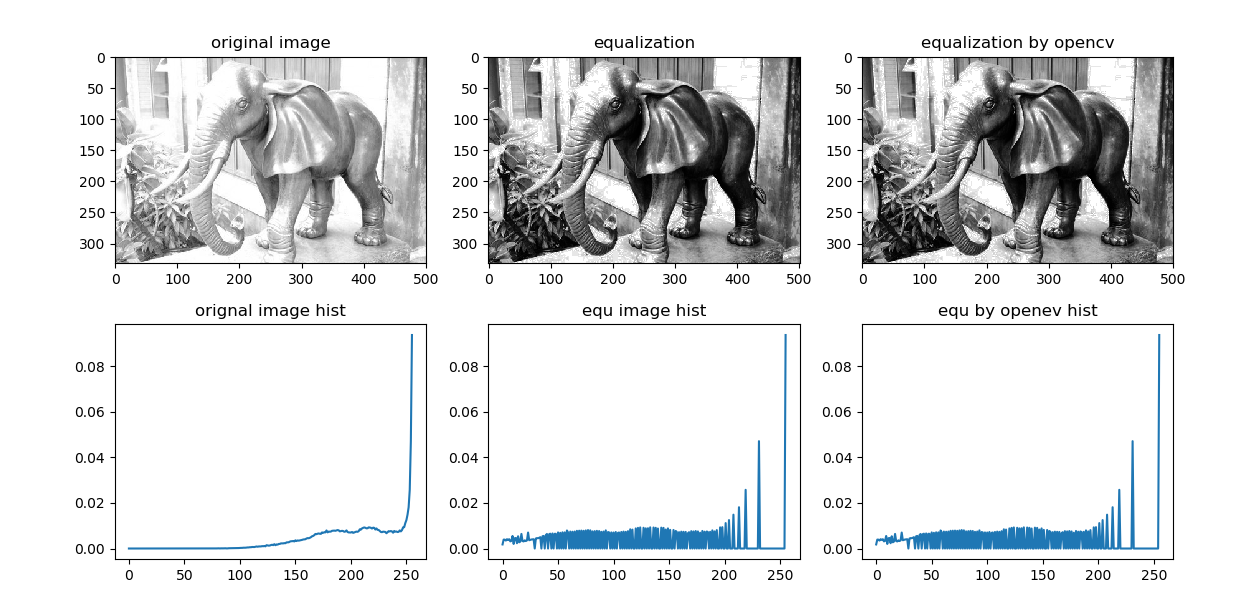
对低对比度的图像进行拉伸和用直方图均值化进行优化。

1. 使用伽马变换进行拉伸，由于低对比度图像偏亮（曝光过度），将其变暗，γ>1，分别取2，3，4，5进行拉伸变换，效果如下。结果完成后，与skimage的伽马变换函数进行效果比对，基本相同。根据显示效果，当γ=4时效果最佳，实验结束。



1. 对图像进行直方图均衡优化

首先计算p(rk),然后求出累计直方图sk，并进行取整操作skInt。建立转换函数s=skInt[rk]然后使用map函数，对原像素值进行转换。效果如下。使用opencv的直方图均衡方法，比较两者差异，效果基本相同，实验结束。



代码如下：

###2019年9月14日 打开一幅低对比度图像，拉伸其图像，直方图均衡。  
#使用skimage开发包 和 根据原理自行开发，并对比显示其效果。  
import cv2 as cv  
import numpy as np  
import matplotlib.pyplot as plt  
from skimage import exposure  
  
  
img = cv.imread("./source/02.lowcontrast.jpg",cv.IMREAD\_GRAYSCALE) #读入做灰色处理  
print(img.shape)  
  
  
  
#使用skimage的伽马变换  
gamma\_img = exposure.adjust\_gamma(img, 5)  
#根据原理开发的伽马变换：对图像拉伸，先需做归一化(/255去除量纲），然后进行变换，最后复原回图像空间（需做无符号转换）  
#曝光过度，调黑。尝试γ取2，3，4，5，检查效果  
stretch2 = np.uint8(np.power(img / 255.0, 2) \* 255.0)  
stretch3 = np.uint8(np.power(img/255.0,3)\*255.0)  
stretch4 = np.uint8(np.power(img/255.0,4)\*255.0)  
stretch5 = np.uint8(np.power(img/255.0,5)\*255.0)  
print(stretch2)  
#对比skimage与自行开发的伽马变换。  
#cv.imshow("dev and skimage",np.hstack((stretch15,gamma\_img)))  
figure,axs =plt.subplots(2,3)  
axs[0,0].set\_title("original image")  
axs[0,0].imshow(img,cmap='gray') #原图  
  
axs[0,1].set\_title("stretch 2")  
axs[0,1].imshow(stretch2, cmap='gray')  
  
axs[0,2].set\_title("stretch 3")  
axs[0,2].imshow(stretch3,cmap='gray')  
  
axs[1,0].set\_title("stretch 4")  
axs[1,0].imshow(stretch4,cmap='gray')  
  
axs[1,1].set\_title("stretch 5")  
axs[1,1].imshow(stretch5,cmap='gray')  
  
axs[1,2].set\_title("package skimage stretch 5")  
axs[1,2].imshow(gamma\_img,cmap='gray')  
  
  
#skimage的直方图均衡函数  
equ = cv.equalizeHist(img)  
#cv.imshow('equalization by opencv', np.hstack((img, equ))) # 并排显示  
  
#按照公式计算：  
#原始图像灰度级统计  
rk = np.bincount(img.ravel(), minlength=256) #图像二维数组先转换为一维，然后使用bincount求出不同像素级数的个数  
prk=rk/img.size #各级像素的概率  
sk=np.zeros(256)  
pos = 0  
sum = 0.0  
for val in prk:  
 sum = sum + val #累计  
 sk[pos] = sum  
 pos = pos+1  
  
#展示处理后的直方图  
#plt.figure()  
#plt.plot(sk)  
  
print(sk,pos,sum)  
  
#sk取整扩展  
skInt = np.uint8(255\*sk+0.5)  
  
print(skInt)  
  
def fn\_change(x):  
 return skInt[x]  
  
equImg2=np.array(list(map(fn\_change,img)),dtype=np.uint8)  
equFigure,equAxs =plt.subplots(2,3)  
equAxs[0,0].set\_title("original image")  
equAxs[0,0].imshow(img,cmap='gray') #原图  
  
equAxs[0,1].set\_title("equalization")  
equAxs[0,1].imshow(equImg2, cmap='gray')  
  
equAxs[0,2].set\_title("equalization by opencv")  
equAxs[0,2].imshow(equ, cmap='gray')  
  
#展示图像直方图  
# plt.hist(img.ravel(), 256, [0, 256])#等价于 bincount先求出每个级数的个数，然后画直方图。  
hist = np.bincount(img.ravel(), minlength=256)/img.size  
equAxs[1,0].set\_title("orignal image hist")  
equAxs[1,0].plot(hist) #原图  
  
hist2 = np.bincount(equImg2.ravel(), minlength=256)/img.size  
equAxs[1,1].set\_title("equ image hist")  
equAxs[1,1].plot(hist2)  
  
hist3 = np.bincount(equ.ravel(), minlength=256)/img.size  
equAxs[1,2].set\_title("equ by openev hist")  
equAxs[1,2].plot(hist3)   
  
plt.show()  
  
cv.waitKey(0)