Colors

زهرا نیازی

اطلاعات گزارش

تاریخ: 13/09/1401

واژگان کلیدی:

چکیده

در این تمرین به بررسی تصاویر رنگی و فضاهای رنگی می پردازیم و تصاویر را در فضاهای رنگی مختلف نشان داده و خاصیت های آنان را بیان می کنیم. همچنین تصاویر رنگی را quantized می کنیم و میزان تغییری که هر quantization بر تصویر اعمال می کند را نیز اندازه خواهیم گرفت.

1-مقدمه

در این فصل با رنگ ها، فضاهای رنگی و پردازش هایی که روی تصاویر رنگی انجام می شود آشنا شده و در ادامه به حل و بررسی چند تمرین برای کار با فضاهای رنگی و کاربرد های آنها می پردازیم.

2-شرح تکنیکال

Color space 1-2

1-1-2

در این تمرین خواسته شده تصویر Lena را به فرمت HSI تبدیل کنیم و اجزای HIS را به صورت تصاویر جداگانه در مقیاس خاکستری نمایش دهیم.

سپس این تصاویر را بررسی میکنیم تا بفهمیم هر یک از اجزای H، S، I چه چیزی را نشان می دهند.

در نهایت تصاویر HSI را با دقت double ذخیره میکنیم.

1-1-2

در این تمرین خواسته شده فضای رنگی جدید (حداقل سه تا) را که در کلاس معرفی نشده است را با جزئیات ارائه دهیم.

LUV که به نام CIELUV نیز شناخته می شود و مشابه LAB است و همچنین بر اساس فضای رنگ ی CIE XYZ است.

مانند LAB، LUV به گونه ای طراحی شده است که از نظر ادراکی یکنواخت باشد، اما از مجموعه متفاوتی از کانال های رنگی استفاده می کند.

فضای رنگ ی LUV از فضای رنگ ی CIE Luv* مشتق شده است:

L* نشان دهنده lightness

u* نشان دهنده محور قرمز-سبز

v* نشان دهنده محور زرد-آبی

فضای رنگی LUV اغلب در برنامههای بینایی کامپیوتری و پردازش تصویر استفاده میشود، زیرا نمایش دقیقتری از رنگ را نسبت به سایر فضاهای رنگی مانند RGB یا HSI ارائه میدهد

* zahra.niazi2001@gmail.com

YUV یک فضای رنگی است که اطلاعات درخشندگی را از اطلاعات کرومینانس جدا می کند. YUV اغلب در فشرده سازی و یدئو و تصویر استفاده می شود زیرا امکان فشـــرده ســـازی موثر اطلاعات رنگ را فراهم می کند. در YUV کانال Y حاوی اطلاعات درخشـــندگی اســـت، در حالی که کلنال ها ی U و V حاوی اطلاعات کرومینانس هستند. از فضای رنگ ی YUV در پخش تلویزیونی نیز استفاده م ی شود، جا یی که از کانال Y برای انتقال تک رنگ و کانال های U و V برای انتقال رنگی اســـتفاده می شود. برخلاف RGB و HSI، YUV یک فضای رنگی از نظر ادراکی یکنواخت نیســـت، بنابراین برای تصـــحیح رنگ و ســـایر کارهایی که نیاز به درک دقیق رنگ دارند، ایده آل نیست.

Quantization 2-2

1-2-2

در این تمرین از ما خواســـته شـــده اجرای کمی ســـازی یکنواخت یک تصـــویر رنگی را انجام دهیم. طبق مراحل زیر پیش میرویم:

1. یک تصویر خاکستری را در یک آرایه بخوانید.
2. تصـــویر کوانتیزه شـــده را در یک آرایه دیگر کمی کنید و ذخیره کنید.
3. MSE و PSNR را بین تصاویر اصلی و کوانتیزه محاسبه کنید.
4. تصویر کوانتیزه شده را نمایش و چاپ کنید.

نکته قابل توجه این اســـت که باید مقادیر ورودی را در محدوده (0256) فرض کنیم، اما میتوان سطح بازسازی را تغییر داد. MSE و PSNR بـه دســـت آمـده بـا L=64,32,16,8 را ذخیره میکنیم و تصـــاویر کوانتیزه شده را با مقادیر L مربوطه نمایش میدهیم.
در انتها کیفیت تصویر را بررسی میکنیم.

2-2-2

در این تمرین از ما خواسته شده برای تصویر Lena، با استفاده از یک کوانتایزر یکنواخت، اجزای R، G و B را به ترتیب به 3، 3 و 2 بیت تبدیل کنیم. سپس تصویر رنگی اصلی و کوانتیزه شده را نمایش داده و تفاوت دقت رنگ را بررسی میکنیم.

3-2-2

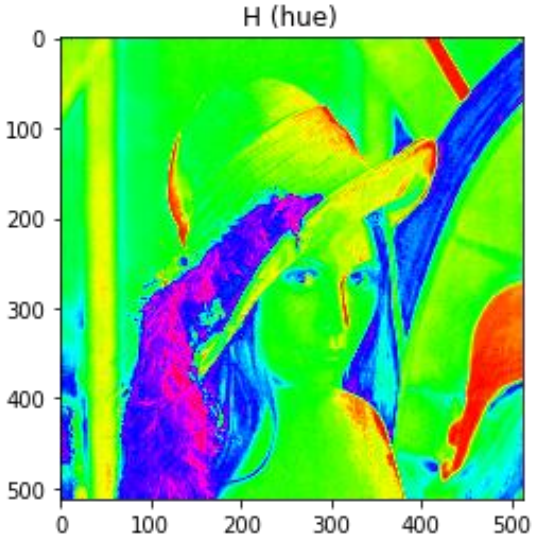
در این تمرین می خواهیم تصویر بابون را روی قالیچه ببافیم. برای انجام این کار، باید تعداد رنگ های تصویر را با حداقل کاهش کیفیت بصری کاهش دهیم. اگر بتوانیم 32، 16 و 8 رنگ مختلف در فرآیند بافت داشته باشیم، باید رنگ تصویر را به این سه حالت خاص کاهش دهیم.

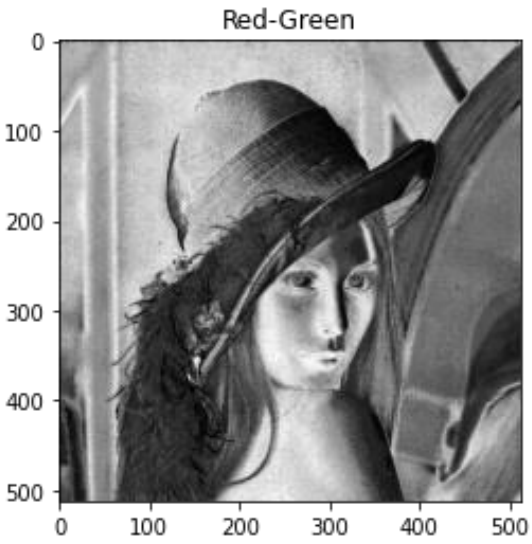
3-نتایج

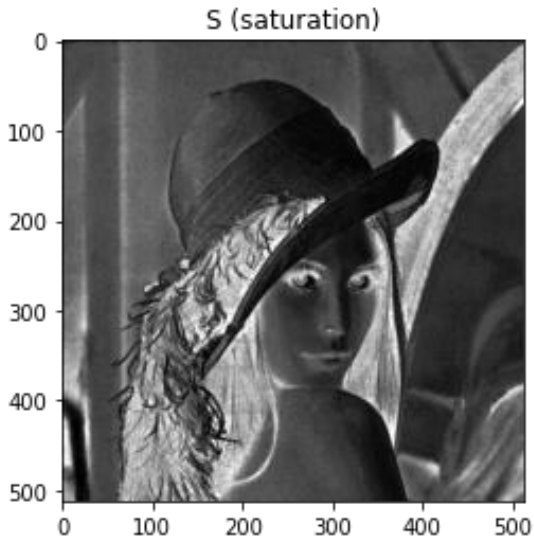
1-3

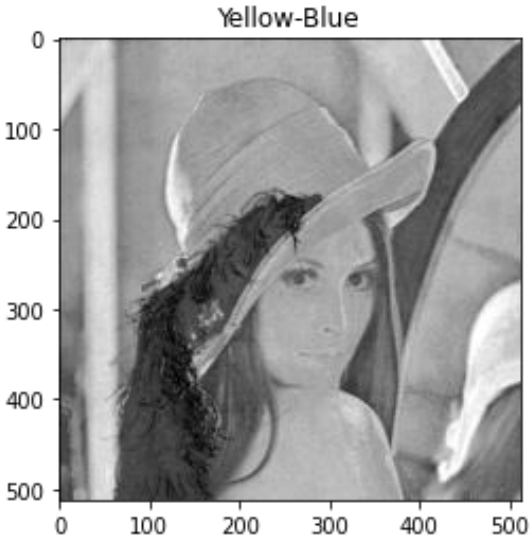
1-1-3

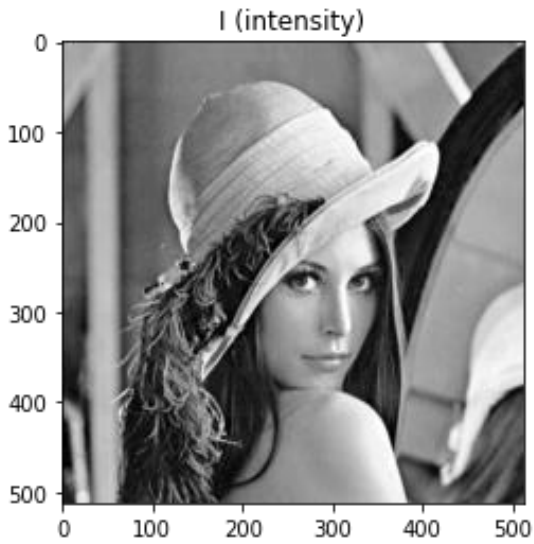
تصاویر زیر نمایش مولفه های Saturation و intensity وhue اســـت. مولفه intensity تمامی اطلاعات به جز اطلاعات مربوط بـه رنـگ هـارا دارد. مولفـه saturation میزان خلوص رنگ در ترکیب با رنگ ســـفید را نشـــان میدهد و مولفه hue فام رنگ غالب را نمایش می دهد.





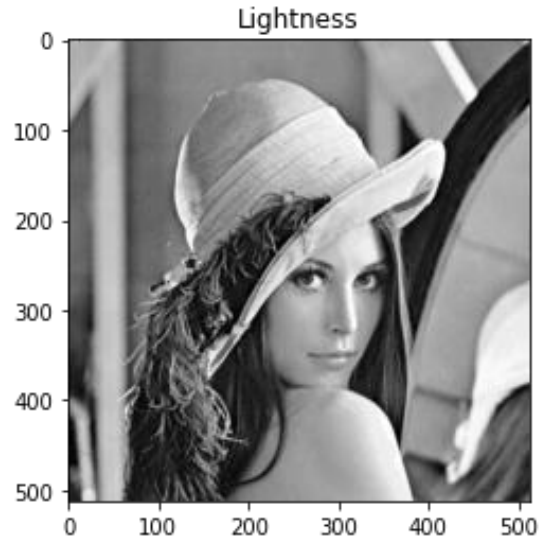




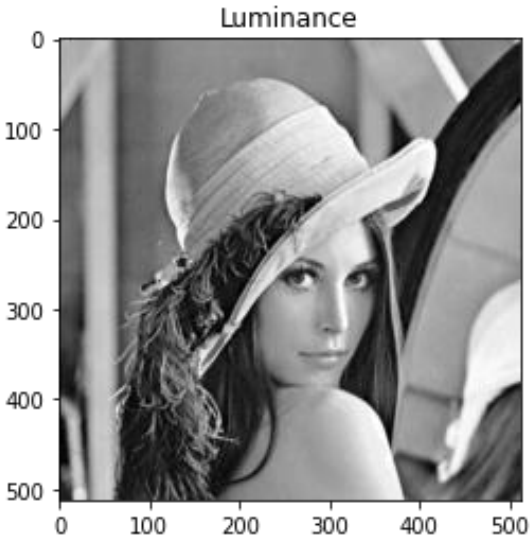


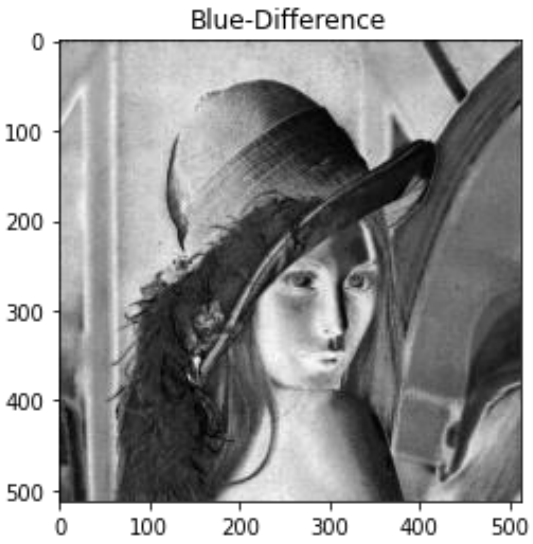
2-1-3

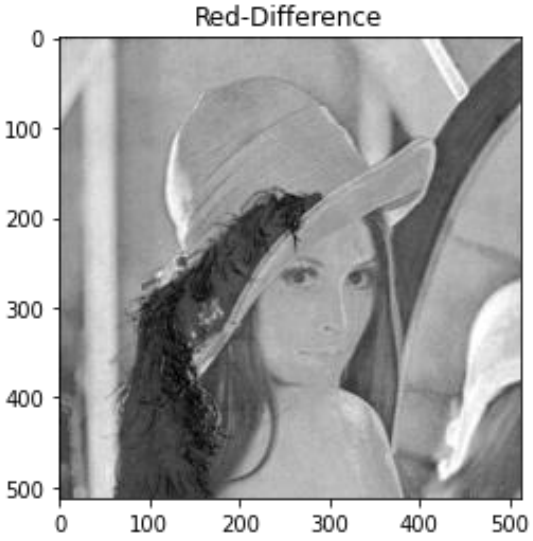
تصاویر زیر نتیجه انتقال تصویر به فضای LUV می باشد:



تصاویر زیر نتیجه انتقال تصویر به فضای YUV می باشد:



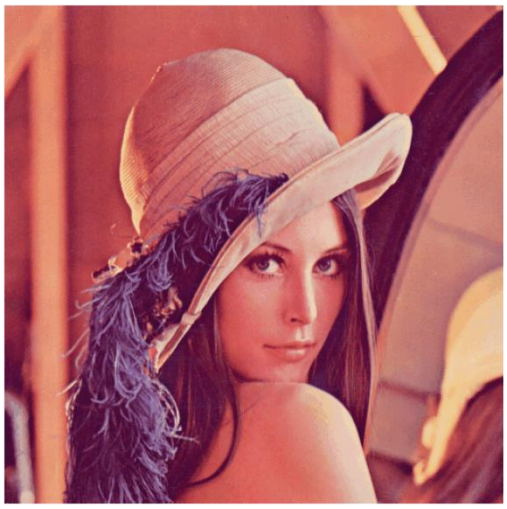




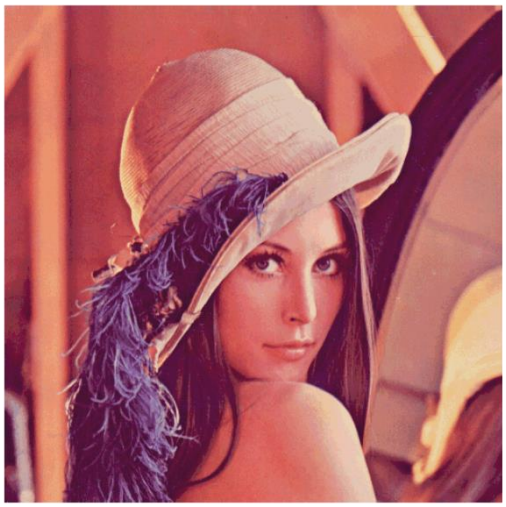
2-3

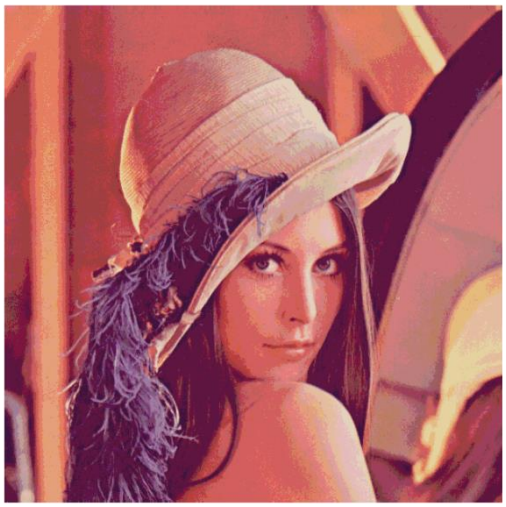
1-2-3

در این تمرین تصویر رنگی ورودی را کوانتیزه کرده و به ترتیب در 64و 32و 16و 8 سطح نمایش میدهیم:



همانطور که انتظار میرفت چشم انسان قادر به تشخیص تفاوت های سطوح 64و32 نیست.





در حالیکه هر چه تعداد سطحها کاهش پیدا کند، کیفیت تصویر کاهش پیدا کرده زیرا جزئیات کمتر می شوند. اما مقادیر mse و psnr برای هر تصویر با تصویر اصلی محاسبه شده است و طبق نتایج هرچه تعاد سطوح رنگی

کوناتیزه شده کمتر میشود مقدار mse بیشتر شده و اختلاف تصویر با تصویر اصلی بیشتر می شود. همچنین با افزایش تعداد سطوح مقدار psnr افزایش می یابد.

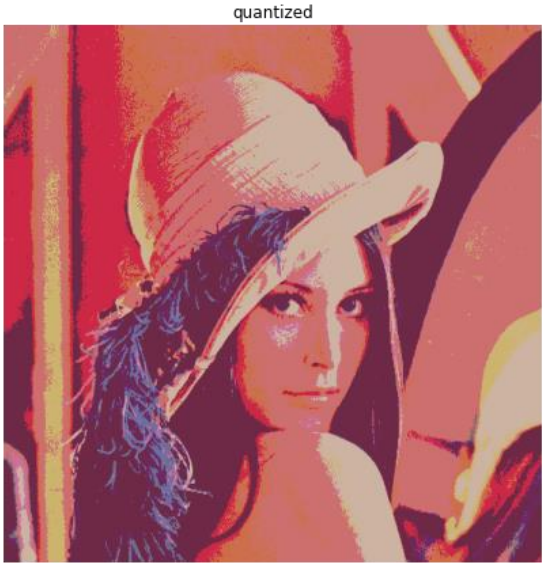
	K	MSE	PSNR
0	64	0.112879	57.604655
1	32	4.061811	42.043607
2	16	16.662619	35.913371
3	8	73.068895	29.493478

2-2-3

از ما خواسته شده که برای تصویر lena مولفه های R,G,B را با 3و3و2 بیت کوانتیزه کنیم. برای این کار هر کانال را با تعداد بیت های زکر شده کوانتیزه میکنیم و در انتها آنها را با هم ترکیب میکنیم.

original_image

quantized



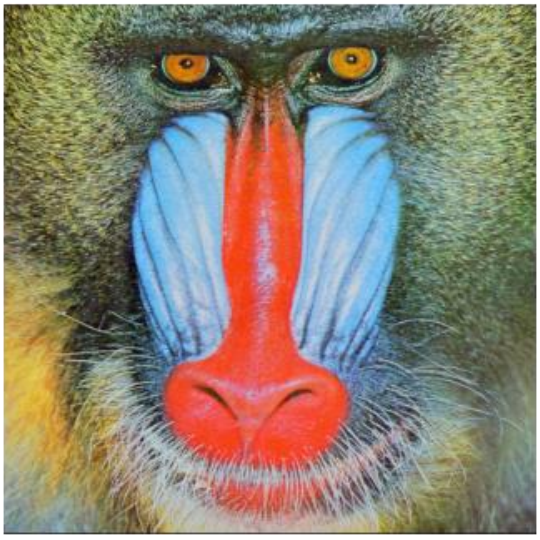
3-2-3

در این تمرین تصویر بابون را با تعداد سطوح مختلفی کوانتیزه کرده و سپس تعداد رنگهای تصویر را کاهش میدهیم.

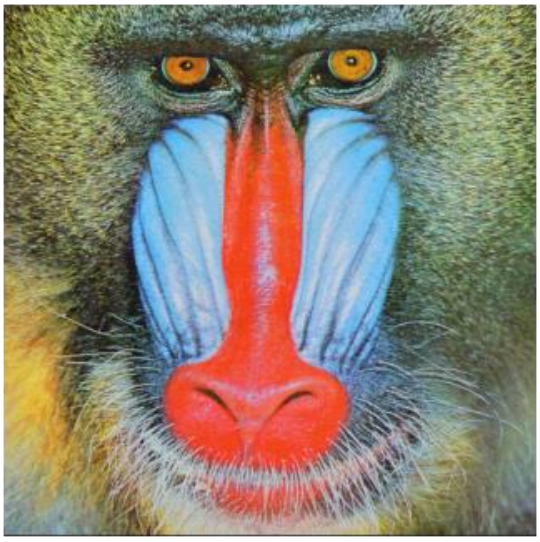
برای این مهم با استفاده از الگوریتم خوشه بندی Kmeans رنگ هایی موجود در تصویر را کلاستربندی میکنیم و سپس تصویر را بر اساس کلاسترها کوانتیزه میکنیم.

	K	MSE
0	32	5.123638
1	16	18.656756
2	8	66.932964

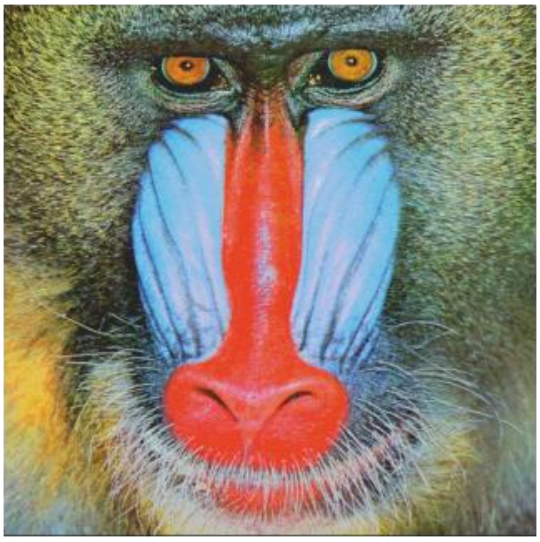
k = 32



k = 16



k = 8



4-Code

https://colab.research.google.com/drive/18RwqsqiYw0uKYIdv9OefM3tcmQFeJMa9?usp=sharing

```
def RGB_to_HSI(img):
  r, g, b = img[:, :, 0], img[:, :, 1], img[:, :, 2]
  r = r / 255.0
  g = g / 255.0
  b = b / 255.0
  i = (r + g + b) / 3
  theta = np.arccos(0.5 * ((r - g) + (r - b)) / np.sqrt((r - g)**2 +
  (r - b) * (g - b)))
  h = np.where(b <= g, theta, 2 * np.pi - theta)
  s = 1 - 3 * np.minimum(np.minimum(r, g), b) / (r + g + b)

  return [h, s, i]


img_luv = cv2.cvtColor(lena, cv2.COLOR_BGR2LUV)
l, u, v = cv2.split(img_luv)


img_yuv = cv2.cvtColor(lena, cv2.COLOR_BGR2YUV)
y, u, v = cv2.split(img_luv)


def quantize_global(x, k):
  k_means = MiniBatchKMeans(k, compute_labels=False)
  k_means.fit(x.reshape(-1, 1))
  labels = k_means.predict(x.reshape(-1, 1))
  q_x = k_means.cluster_centers_[labels]
  q_img = np.uint8(q_x.reshape(x.shape))
  return q_img



def quantize_channels(x, k):
  quantized_x = x.copy()
  for d in range(len(k)):
    channel = x[:, :, d].copy()
    k_means = MiniBatchKMeans(k[d], compute_labels=False)
    k_means.fit(channel.reshape(-1, 1))
    labels = k_means.predict(channel.reshape(-1, 1))
    quantized_x[:, :, d] = np.uint8(k_means.cluster_centers_[labels]).re
shape(channel.shape)
  return quantized_x
```

```
def get_region_index(color_value):
    eight_regions = [[0,31], [32,63], [64,95], [96,127], [128,159], [160
,191], [192,223], [224,255]]
    for index, region_value in enumerate(eight_regions):
        if color_value >= region_value[0] and  color_value <= region_val
ue[1]:
            return index



def mse(imageA, imageB):
  return np.square(imageA.astype('int16') -
                   imageB.astype('int16')).mean()



def PSNR(original, compressed):
  MSE = mse(original, compressed)
  if MSE == 0:
      return 100
  max_pixel = 255.0
  psnr = 20 * log10(max_pixel / sqrt(MSE))
  return psnr




q64 = quantize_global(lena, 64)
q32 = quantize_global(lena, 32)
q16 = quantize_global(lena, 16)
q8 = quantize_global(lena, 8)



pandas.DataFrame(columns=('K','MSE','PSNR'),
                 data=[('64', mse(q64, lena), PSNR(q64, lena)),
                       ('32', mse(q32, lena), PSNR(q32, lena)),
                       ('16', mse(q16, lena), PSNR(q16, lena)),
                       ('8', mse(q8, lena), PSNR(q8, lena))
                       ])
quantized_x = quantize_channels(lena,[3,3,2])
imshow(lena, quantized_x, title=['original_image','quantized'], figsize=
16)
```

```
q32 = quantize_global(baboon, 32)
q16 = quantize_global(baboon, 16)
q8 = quantize_global(baboon, 8)

imshow(q32, q16, q8, title=['k = 32', 'k = 16', 'k = 8'], figsize=20)

pandas.DataFrame(columns=('K','MSE'),
                 data=[('32', mse(q32, baboon)),
                       ('16', mse(q16, baboon)),
                       ('8', mse(q8, baboon))
                       ])
```