سید محمد طاها طباطبایی – تمرین سری دوم

9812762838

چکیده:

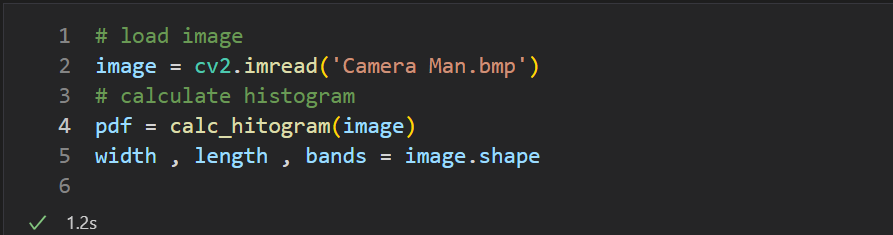
در بخش اول، به بررسی و تحلیل هیستوگرام و متعادل سازی، تبدیل و اعمال توابعی که شدت روشنایی پیکسل ها را تحت تاثیر قرار می‌دهند، پرداختیم. در بخش دوم، به طور مشخص، به مقایسه تاثیر متعادل سازی کلی بر روی تصویر camera man پرداخته ایم.

در پارت آخر، متعادل سازی محلی را روی 4 نمونه تصویر مختلف و با سایز پنجره های متفاوت اعمال کردیم، تا بتوانیم تاثیر ابعاد پنجره مختلف و همچنین تاثیر متعادل سازی در تصاویر با نرخ جزئیات متفاوت را بررسی کنیم.

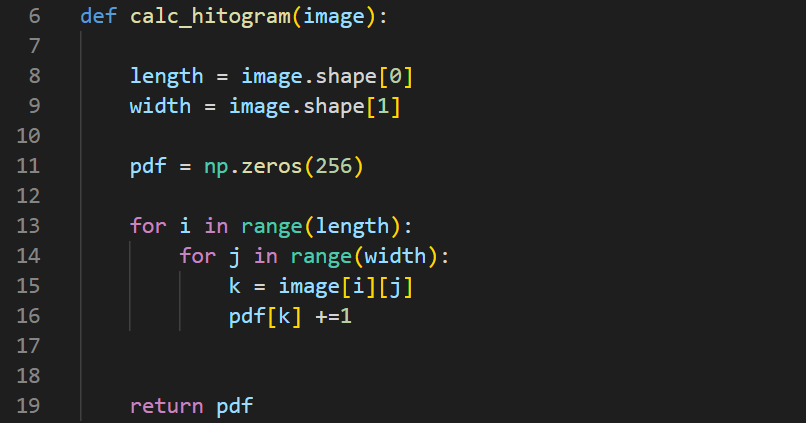
2.1.1

توضیح فنی:

برای محاسبه هیستوگرام یک عکس مطابق عکس زیر، ابتدا با کمک لایبرری OpenCV ، عکس مورد نظر را لود می‌کنیم.



سپس مطابق خط 4، با استفاده از تابع ای که برای محاسبه هیستوگرام نوشته ام، هیستوگرام عکس را به‌دست می‌آوریم. تابع calc\_histogram به شکل زیر تعریف می‌شود:



در این تابع، عکس مورد نظر به عنوان ورودی تابع استفاده می‌شود. در دو حلقه تو در تو، فراوانی رخ داد هر پیکسل که مقداری بین 0 تا 255 دارد را، در یک آرایه به نام pdf ذخیره می‌کنیم. طول این آرایه 256 است، و اندیس k این آرایه، نشان دهنده فراوانی پیکسل هایی با مقدار k است.



در گام آخر، مطابق قطعه کد بالا، با یک نمودار از نوع stem و با استفاده از کتابخانه matplotlib، نمودار هیستوگرام مربوط به عکس camera man را رسم می‌کنیم. در محور y این نمودار، مقادیر فراوانی یا همان pdf را جایگزاری می‌کنیم، و در محور x اعداد 0 تا 255.

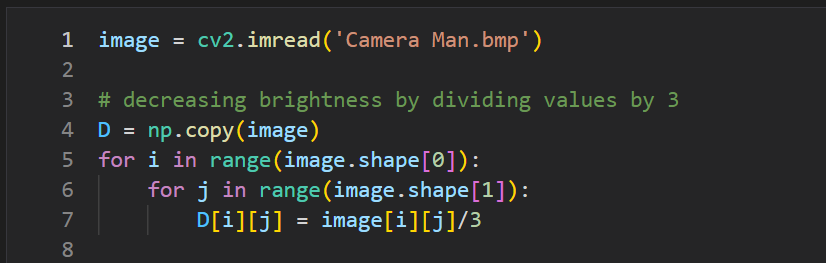
نتیجه:

با اجرای این سلول ها، نمودار هیستوگرام عکس camera man را رسم می‌کنیم. در سه بخش این هیستوگرام تراکم پیکسل های را مشاهده می‌کنیم. بخش اول، از پیکسل هایی با مقدار 20 تا اندکی پایین تر از 10، که می‌توان دلیل آن را، وجود کت سیاه رنگ در تصویر باشد. همچنین یک رنج از پیکسل ها با مقادیر بین 100 تا 140 تا می‌توان به ناحیه ای که زمین را شامل می‌شود نسبت داد. رنج 150 تا 190 نیز می‌تواند نمایشگر بخش آسمان و فراوانی پیکسل های مربوط به این بخش باشد.

2.1.1.1

توضیح فنی:

در قطعه کدی مشابه تصویر زیر، مقدار هر پیکسل را بر 3 تقسیم می‌کنیم. با اینکار انتظار داریم، تصویر تاریک‌تر شود. برای اینکار یک کپی از تصویر اصلی تهیه می‌کنیم و نام آن را D می‌گذاریم.

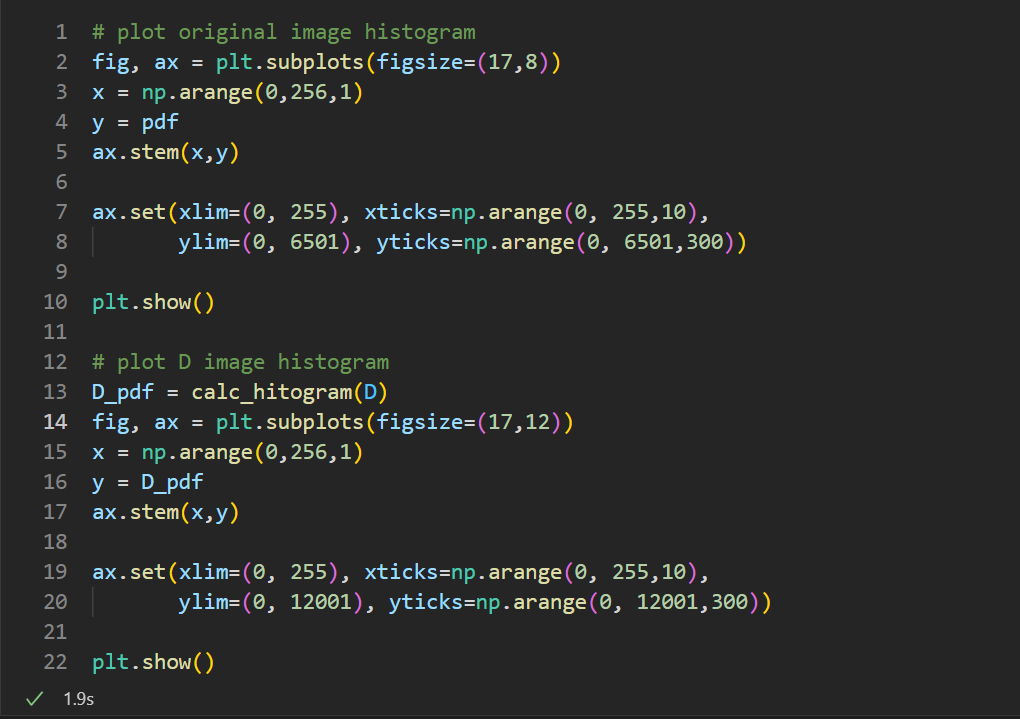


نتیجه:

در بخش بعد بررسی می‌شود.

2.1.1.2

توضیح فنی:



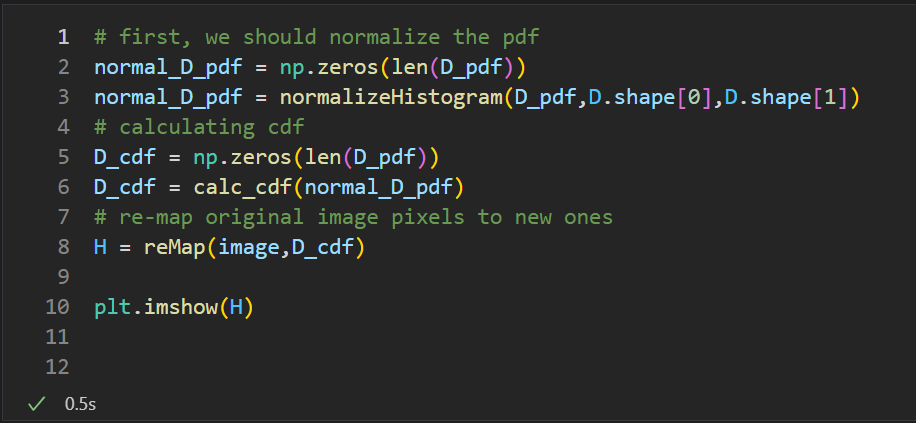
مطابق تصویر بالا، نمودار هیستوگرام عکس اصلی و عکس D را رسم می‌کنیم. هیستوگرام عکس D در خط 13 محاسبه شده است. اجرای قطعه کد بالا، دو نمودار را به ترتیب ذکر شده رسم می‌کند.

نتیجه:

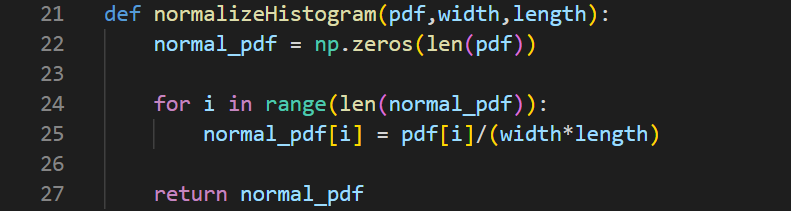
با اجرای سلول این بخش، در نمودار اول هیستوگرام عکس اصلی را رسم می‌کنیم که در بخش قبل نیز مشاهده شد. در نمودار دوم که مربوط به هیستوگرام عکس D است، تفاوت فاحشی بین فراوانی پیکسل ها با عکس اصلی داریم. فراوانی پیکسل های بخش های روشن تصویر، به طور معنی داری، کاهش یافته، و فراوانی پیکسل های بخش های تاریک‌تر تصویر، بسیار بیشتر شده است. به طور مثال، رنج پیک 150 تا 190 در عکس قبلی، به حدود 50 تا 65 مپ شده است و فراوانی این پیکسل ها بسیار زیادتر از عکس اول شده است.

2.1.1.3

توضیح فنی:

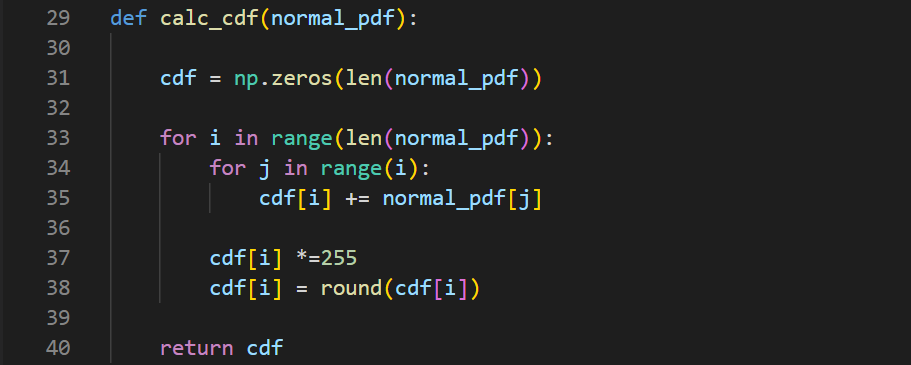


برای اجرای روال متعادل سازی هیستوگرام، در ابتدا نیاز داریم تا طبق الگوریتم ارائه شده در درس، هیستوگرام فعلی را نرمالایز کنیم. برای اینکار یک تابع به نام normalizeHistogram نوشته ایم.



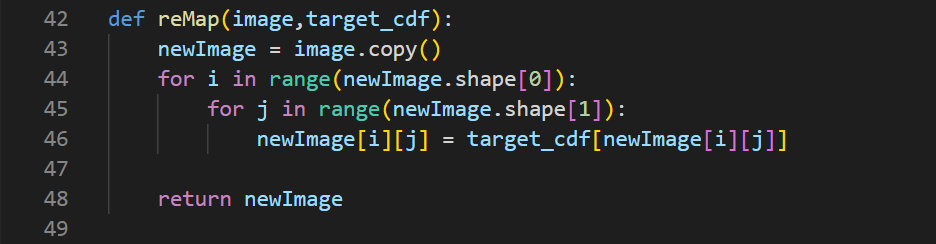
در این تابع، مقادیر pdf را به رنج بین 0 تا 1 مپ می‌کنیم. چون ماکسیمم مقدار فراوانی برابر حاصل ضرب طول در عرض تصویر است، برای تقسیم کردن در مخرج از این عدد استفاده می‌کنیم.

در گام دوم، مطابق الگوریتم به محاسبه cdf (تابع چگالی) می‌پردازیم. برای اینکار تابع calc\_cdf را تعریف می‌کنیم.



در این تابع، به ازای مقدا i ام از آرایه نرمال شده، تمام اندیس های 0 تا i از آرایه نرمال شده را با هم جمع می‌کنیم. طبق فرمول محاسبه cdf، در حلقه بیرونی، که در واقع انتهای سیگمای جمع مقادیر نرمالایز شده تا آن اندیس است، مقدار آن اندیس را در 255 ضرب می‌کنیم. در انتها چون این مقدار ممکن است یک عدد اعشاری باشد، آن را به مقدار صحیح نزدیک تر به آن گرد می‌کنیم.

در گام نهایی نیاز است تا با توجه به تابع تبدیل به دست آمده، مقادیر پیکسل های عکس را آپدیت کنیم. برای اینکار، از یک تابع به نام reMap نوشته ایم.



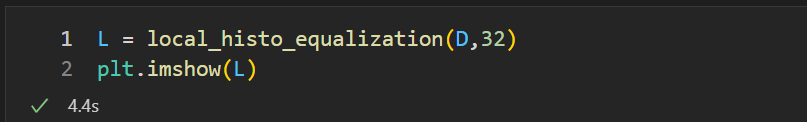
این تابع با دریافت عکس هدف و cdf مورد نظر، پیکسل های عکس را با مقادیر جدید بر اساس cdf داده شده، آپدیت می‌کند.

نتیجه:

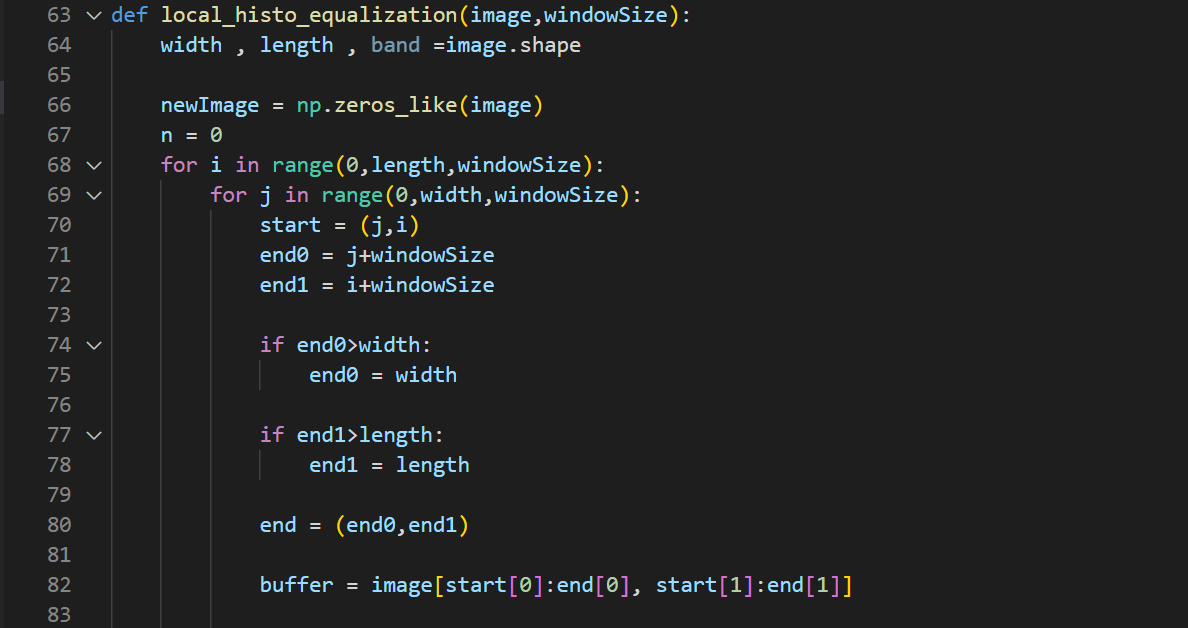
اگر نتیجه این تبدیل را مشاهده کنیم، می‌بینیم که عکس دارای قسمت های بسیار سفید است، که به دلیل وجود شیب زیاد در تابع تبدیل به دلیل وجود قله هایی از فراوانی پیکسل، در هیستوگرام عکس ورودی است. یک روش حل این مشکل، برش زدن هیستوگرام است، که چون در صورت سوال رفع این مشکل خواسته نشده، رفع مشکل پیاده‌سازی نشده است. روش دیگر حل این مشکل، استفاده از متعادل سازی محلی است، که در بخش بعدی پیاده سازی شده است.

2.1.1.4

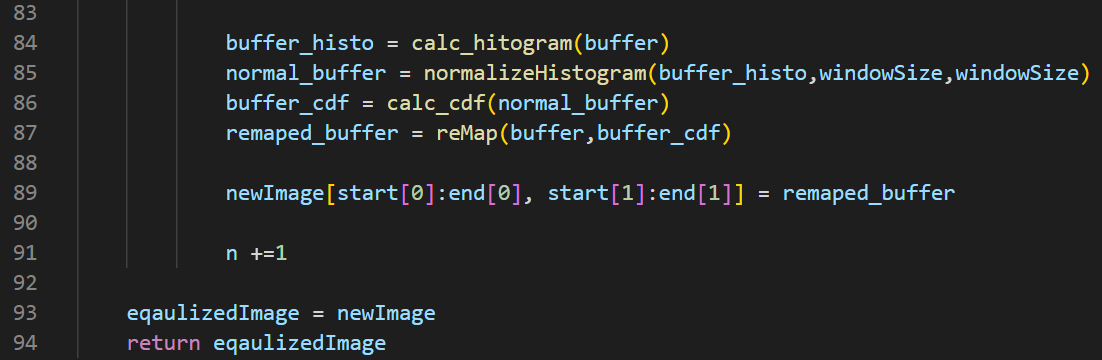
توضیح فنی:



برای پیاده‌سازی متعادل سازی محلی هیستوگرام، یک تابع پیاده سازی کرده ایم.



ورودی این تابع، یک عکس و ابعاد پنجره مورد نظر برای بخش‌بندی تصویر بر اساس این پنجره و محاسبات به صورت محلی در این پنجره است. مطابق شکل، ابتدا براساس اندازه پنجره، روی عکس حرکت می‌کنیم و یک برش با اندازه windowsize\*windowsize از عکس را انتخاب می‌کنیم. متغییر بافر، یک متغییر میانی است که ابتدا محاسبات را برای هر برش از عکس، روی این تصویر انجام می‌دهیم، سپس نتایج را به متغییر مورد نظر برای عکس جدید منتقل می‌کنیم.



در ادامه، مطابق حالت عادی متعادل سازی، برای هر بخش، روال متعادل سازی را روی برش انتخاب شده، اعمال می‌کنیم. در انتهای حلقه، بافر ریمپ شده را، در یک آرایه با نام newImage که درواقع عکس تولیدی جدید است، جایگزاری می‌کنیم.

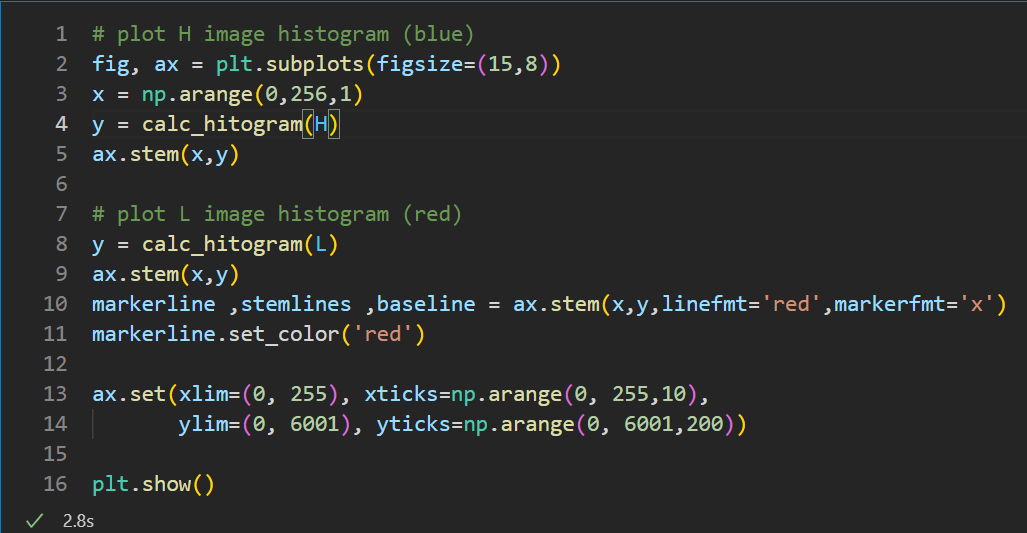
نتیجه:

با مشاهده عکس تولیدی در این مرحله، می بینیم که عکس تولیدی در بخشی نقاط حالت پنجره پنجره دارد. دلیل این اتفاق، این است که در پنجره هایی که از نقاط یکدست تصویر نمونه برداری شده است، فقط فراوانی برخی مقادیر خاص از رنج 0 تا 255 زیاد است، و فراوانی بقیه مقادیر نزدیک به صفر است، و وجود این پیک ها متعادل سازی را کمی با مشکل مواجه می‌کند. اما در بخش هایی که پنجره ها، از قسمت هایی نمونه برداری کرده‌اند که شامل جزیئات عکس است، این حالن پنجره پنجره شدن کمتر دیده می‌شود و تا حدود زیادی، عملکرد محلی سازی بهتر از متعادل سازی کلی است، چون باعث می‌شود تا کنتراست بهتر تغییر کند و سایه ها و لبه ها نیز مشخص تر هستند.

2.1.1.5

توضیح فنی:

هیستوگرام های H و L را محاسبه و رسم می‌کنیم. میله های آبی رنگ روی نمودار، نشان دهنده مقادیر هیستوگرام تصویر H و میله های قرمز مربوط به تصویر L است.



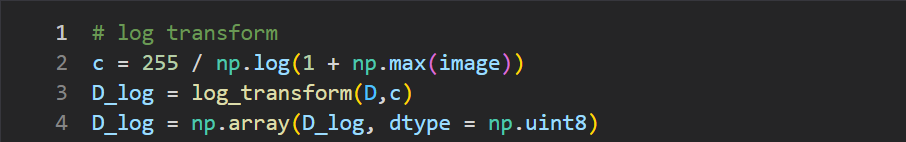
نتیجه:

مطابق نمودار های رسم شده، میبینیم که هیستوگرام تصویر L ، نسبت به تصویر H از پراکندگی بیشتری برخوردار است. مطابق انتظار، هیستوگرام تصویر H، به دلیل اینکه شامل دو ناحیه سفید و سیاه است، به همین شکل باشد، یعنی در رنج بالاتر از 250 و رنج 50 تا 60 ، فراوانی مقادیر بسیار زیاد است و در بقیه مقادیر، فراوانی ها کم یا صفر است. برعکس در تصویر L ، نموداری داریم که تقریبا فراوانی پراکنده ای دارد، و نمی توان یک قله مشخص و برجسته را در آن مشخص کرد، علی‌رغم اینکه در برخی مقادیر، یک پرش و افزایش فراوانی، نسبت به مقادیر همسایه خود دارد، که قابل چشم پوشی است.

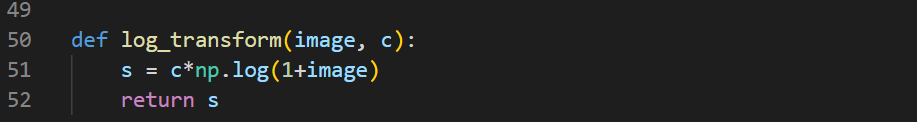
2.1.1.6

توضیح فنی:

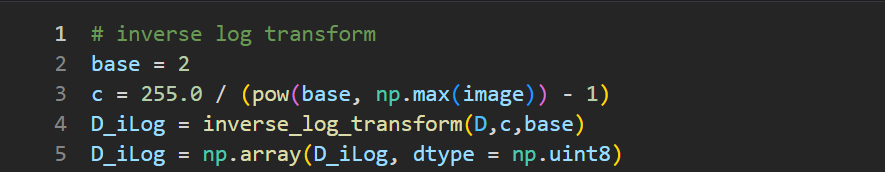
در سلول اول این بخش، تبدیل لگاریتمی مطابق فرمول ارائه شده در اسلاید درس پیاده سازی شده است. در ادامه نیز تصویر حاصل و هیستوگرام آن رسم شده است.



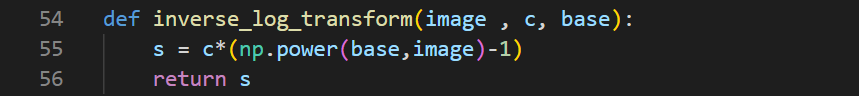
پیاده‌سازی تابع log\_transform به شکل زیر است:



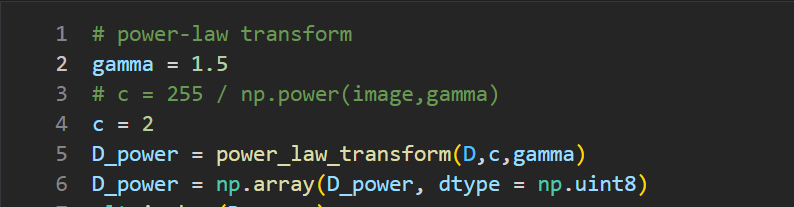
در سلول دوم، تبدیل معکوس لگاریتمی پیاده‌سازی شده است.



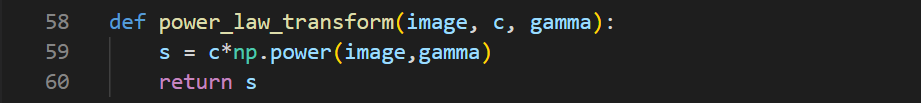
پیاده‌سازی inverse\_log\_transform:



در سلول سوم، تبدیل نمایی پیاده‌سازی شده است.



پیاده‌سازی power\_law\_transform:

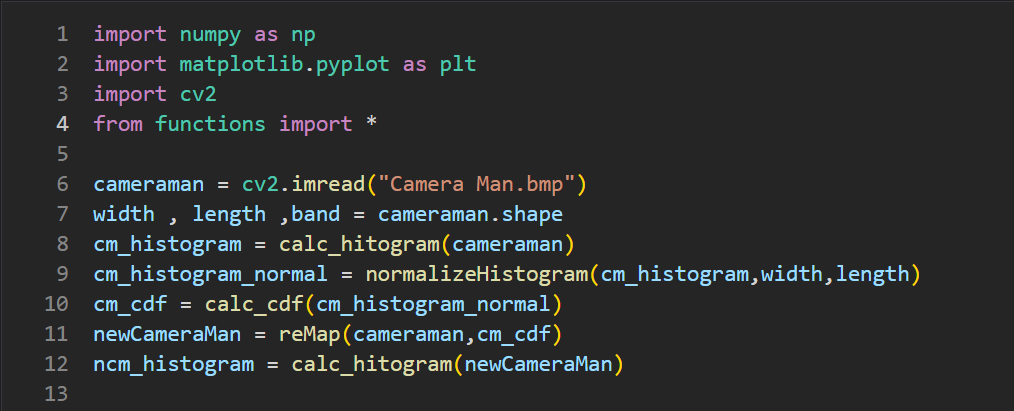


نتیجه:

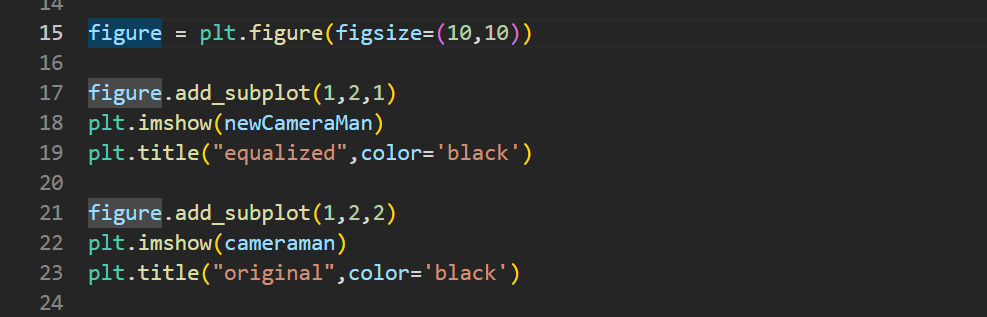
مقادیر پارامتر ها برای توابع، با کمک و مطالعه [این لینک](https://medium.com/alphavision/image-enhancement-46f4b7fda5b4) و [این لینک](https://subscription.packtpub.com/book/application-development/9781784391454/1/ch01lvl1sec16/logarithmic-transformations) انتخاب شده است.

2.1.2

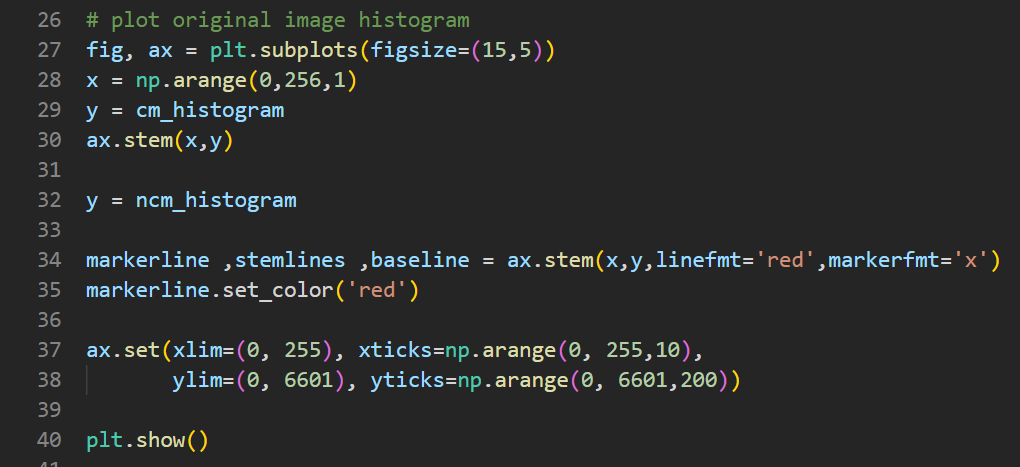
توضیح فنی:



مطابق تصویر و روالی که در قسمت 2.1.1.3 نیز توضیح داده شده، عکس camera man را لود کردیم و مراحل متعادل سازی را روی آن اجرا کرده ایم. در ابتدا هیستوگرام عکس با تابع calc\_histogram محاسبه شده، سپس با تابع normalizeHistogram، هیستوگرام به‌دست آمده را نرمالایز کرده ایم. در گام بعد، cdf را با تابع calc\_cdf محاسبه کرده ایم. در نهایت نیز بر اساس cdf محاسبه شده، عکس جدید camera man را ساخته ایم. در گام نهایی نیز هیستوگرام عکس جدید را حساب کرده ایم.



قطعه کد نمایش عکس اصلی و عکس متعادل شده.



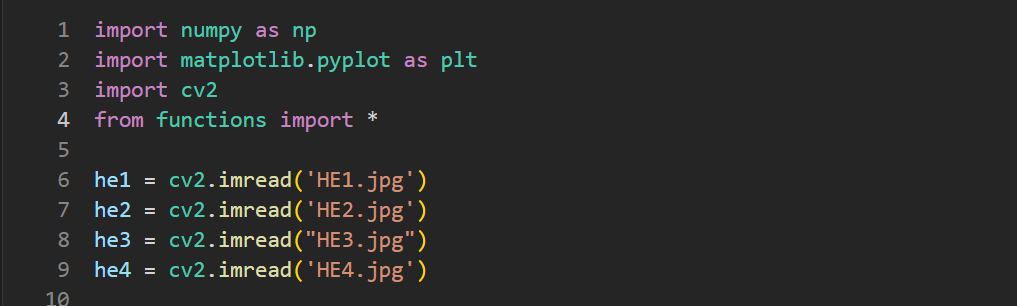
قطعه کد رسم نمودار هیستوگرام عکس اصلی و عکس جدید.

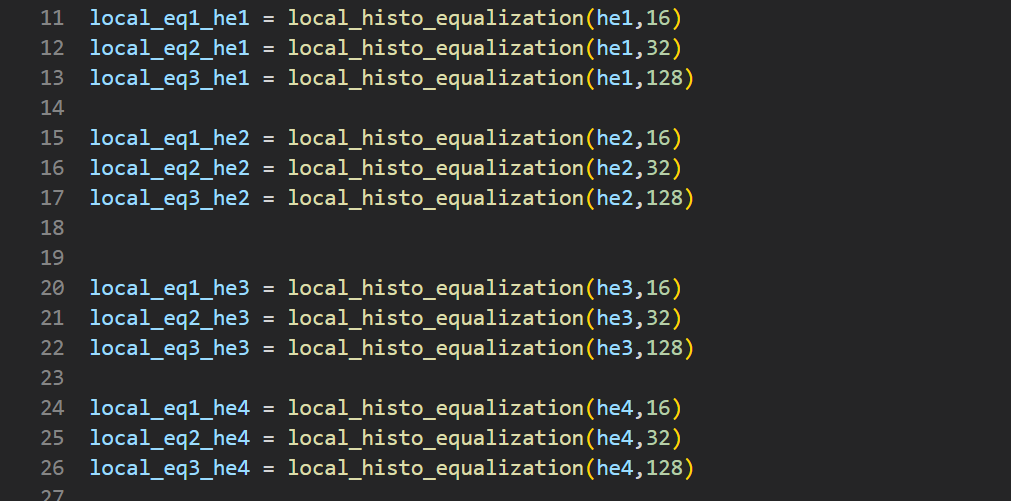
نتیجه:

مطابق انتظار، کنتراست تصویر بهبود یافته است، پراکندگی فراوانی در نمودار متعادل شده بیشتر است و تفاوت بین لبه های سباه و سفید واضح تر شده است.

2.2.1

توضیح فنی:

قطعه کد لود کردن تصاویر.

قطعه کد پردازش تصاویر به صورت محلی، به ترتیب با پنجره هایی با سایز 16،32 و 128.

در ادامه قطعه کد این سلول نیز تصاویر متعادل شده، نمایش داده شده اند.

نتیجه:

در بحث تاثیر ابعاد پنجره، در پنجره های بسیار کوچک، عکس یک حالت پنجره-پنجره شدن پیدا می‌کند، و چون هر پنجره به مقادیر 0 تا 255 اسکیل می‌شود، در هر پنجره کوچک، نقاط بسیار سفید و بسیار تیره پدید می‌آید، که در حالی که برای تصاویر شلوغ، شاید باعث کنتراست بهتر شود، اما قطعا برای نواحی یکدست(فرکانس تغییرات کم) باعث پدیدار شدن حالت نویزی می‌شود.

در سمت مقابل، پنجره بزرگتر، شاید اثر نویزی کمتری داشته باشد، اما در نواحی که تغییرات بسیار شدید است، نسبت به پنجره با ابعاد وچکتر، کنتراست کمتری ایجاد می‌کند. بنابر توضیح بالا، باید با توجه به هر تصویر، تصمیم گرفت که از چه سایز پنجره ای برای متعادل سازی محلی استفاده کرد. با توجه به تصاویر داده شده، سایز پنجره 128 یا 64 حالت مناسبی است، که هم کنتراست خوبی ایجاد می‌کند، و هم عکس حالت نویزی پیدا نمی‌کند.

2.1.3

توضیح فنی:

تابع histeq ، با کمک متعادل‌سازی هیستوگرام، شدت روشنایی پیکسل های تصویر را تغییر می‌دهد، در حالی که تابع imadjust ، شدت روشنایی پیکسل ها را، فارغ از متعادل‌سازی هیستوگرام انجام می‌دهد.

نتیجه:

ندارد