برای حل این سوال نیاز داریم که histogram matching انجام دهیم. بدین صورت که histogram ابتدایی تصویر را بدست می آوریم که با نام source_hist.png موجود است. حال میخواهیم که این هیستوگرام حالت یکنواخت تری داشته باشد تا contrast عکس بهبود بخشیده شود و درنتیجه قسمت های تیره عکس با کیفیت مطلوب تری دیده خواهد شد. بدین منظور نیاز به یک هیستوگرام مقصد(target) داریم. هیستوگرام مقصد را تابع یکنواخت در نظر میگیریم. بدین ترتیب cdf مقصد به راحتی بدست می آید.

اگر بخواهیم این کار را روی عکس رنگی RGB انجام دهیم نتیجه مطلوب نخواهد بود. در نتیجه تصمیم گرفتم که ابتدا عکس را به HSV ببرم و فقط روی لایه V کار کنم. بدین ترتیب لایه V را با کمک هیستوگرام آن بهبود می بخشیم و دوباره عکس را برای خروجی گرفتن به فرمت RGB برمیگردانیم.

حال نیاز داریم که یک تابع پیدا کنیم که به ما بگوید هر i از هیستوگرام اول باید به چه j در هیستوگرام دوم نظیر شود. بدین منظور اقدام به محاسبه تابع های توزیع تجمعی هیستوگرام های مبدا و مقصد می کنیم. حال با توجه به همان روشی که در کلاس گفته شد، به ازای هر i در cdf مبدا، نزدیک ترین j در cdf مقصد را محاسبه می کنیم. این کار را با استفاده از باینری سرچ انجام می دهیم که سریعتر انجام شود(چون cdf صعودی است میتوان این کار را کرد.)

بعد از این ما یک مپ خواهیم داشت که می گوید هر intensity در لایه V عکس اولیه به چه intensity خواهد رفت. بدین ترتیب این تغییر را روی لایه V انجام می دهیم و سپس دوباره ۳ لایه را کنار هم می آوریم و عکس را RGB میکنیم. سپس هیستوگرام آن را نمایش می دهیم که با نام result_hist.png در بین فایل ها موجود است. مشاهده می شود که بسیار یکنواخت تر از هیستوگرام عکس ابتدایی است.

همچنین در عکس result هم قسمت های تاریک بسیار بهتر دیده می شوند و قسمت های دیگر هم آسیب کمی دیده اند و به نظرم نتیجه قابل قبولی است.

در کنار فایل اصلی f02.py یک فایل helper.py قرار دارد که تابع هایی که از آنها استفاده کرده ام را در آنجا نوشته ام تا به خوانایی کدم کمک کند.