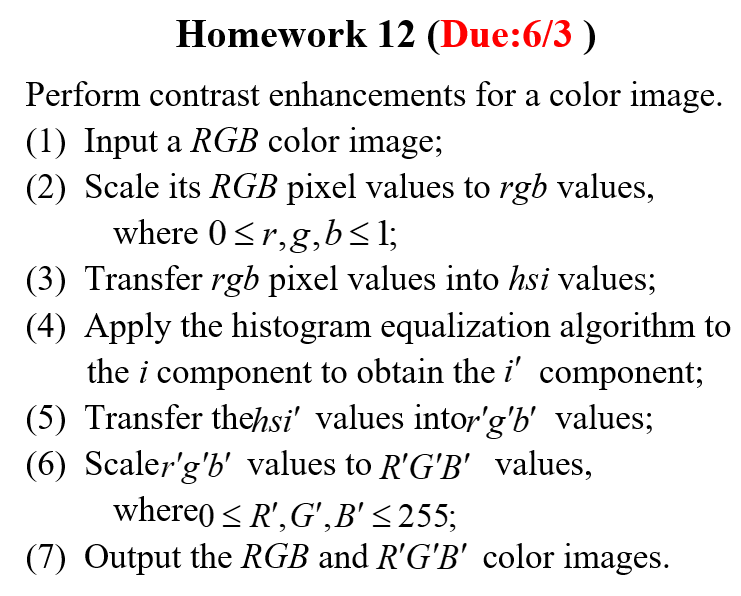
**Problem Statement：**

****

**Input:**

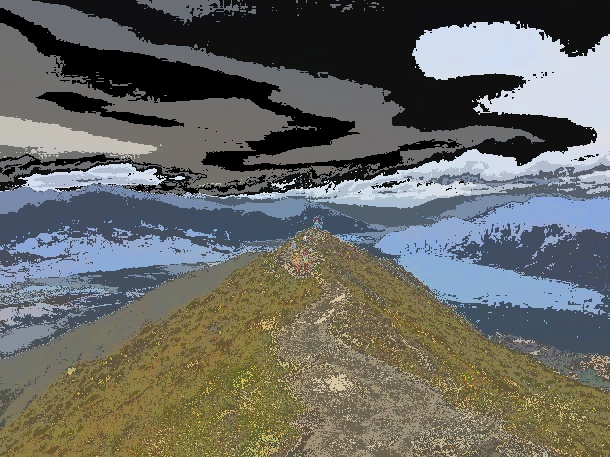


**output:**

***RGB:***



***R’G’B’:***



**Source Code：**

**1.hw12.py:**

**import cv2**

**import pandas as pd**

**import rgbtohsi\_2**

**import numpy as np**

**import math**

**def he\_i(img):**

**row=img.shape[0]**

**col=img.shape[1]**

**for i in range(row):**

**for j in range(col):**

**if img[i,j]==0:**

**img[i,j]=round((15/row\*col)\*sumn[0])**

**elif img[i,j]==1:**

**img[i,j]=round((15/row\*col)\*sumn[1])**

**elif img[i,j]==2:**

**img[i,j]=round((15/row\*col)\*sumn[2])**

**elif img[i,j]==3:**

**img[i,j]=round((15/row\*col)\*sumn[3])**

**elif img[i,j]==4:**

**img[i,j]=round((15/row\*col)\*sumn[4])**

**elif img[i,j]==5:**

**img[i,j]=round((15/row\*col)\*sumn[5])**

**elif img[i,j]==6:**

**img[i,j]=round((15/row\*col)\*sumn[6])**

**elif img[i,j]==7:**

**img[i,j]=round((15/row\*col)\*sumn[7])**

**elif img[i,j]==8:**

**img[i,j]=round((15/row\*col)\*sumn[8])**

**elif img[i,j]==9:**

**img[i,j]=round((15/row\*col)\*sumn[9])**

**elif img[i,j]==10:**

**img[i,j]=round((15/row\*col)\*sumn[10])**

**elif img[i,j]==11:**

**img[i,j]=round((15/row\*col)\*sumn[11])**

**elif img[i,j]==12:**

**img[i,j]=round((15/row\*col)\*sumn[12])**

**elif img[i,j]==13:**

**img[i,j]=round((15/row\*col)\*sumn[13])**

**elif img[i,j]==14:**

**img[i,j]=round((15/row\*col)\*sumn[14])**

**elif img[i,j]==15:**

**img[i,j]=round((15/row\*col)\*sumn[15])**

**else:**

**break**

**img=img\*16**

**return img**

**n=[]**

**sumn=[0]\*16**

**img = cv2.imread('newzeland.jpg')**

**hsi = rgbtohsi\_2.rgbtohsi(img)**

**hsi1=(hsi[:,:,2]/16).astype(int)**

**hsi2=pd.Series(hsi1.flatten())**

**for i in hsi2.value\_counts().sort\_index():**

**n.append(i)**

**for i in range(16):**

**for j in range(i+1):**

**sumn[i]+=n[j]**

**hsi[:,:,2]=he\_i(hsi1)**

**output=rgbtohsi\_2.hsitorgb(hsi)**

**output = output.astype(np.uint8)**

**cv2.imshow('input',img)**

**retval=cv2.imwrite('output.jpg',output)**

**cv2.imshow('output',output)**

**cv2.waitKey(0)**

**cv2.destroyAllWindows()**

**2.rgbtohsi\_2.py:**

**import cv2**

**import numpy as np**

**import math**

**def rgbtohsi(rgb\_lwpImg):**

**rows = int(rgb\_lwpImg.shape[0])**

**cols = int(rgb\_lwpImg.shape[1])**

**B, G, R = cv2.split(rgb\_lwpImg)**

**# 歸一化到[0,1]**

**B = B / 255.0**

**G = G / 255.0**

**R = R / 255.0**

**hsi\_lwpImg = rgb\_lwpImg.copy()**

**H, S, I = cv2.split(hsi\_lwpImg)**

**for i in range(rows):**

**for j in range(cols):**

**num = 0.5 \* ((R[i, j] - G[i, j]) + (R[i, j] - B[i, j]))**

**den = np.sqrt((R[i, j] - G[i, j]) \*\* 2 + (R[i, j] - B[i, j]) \* (G[i, j] - B[i, j]))**

**theta = float(np.arccos(num / (den+0.00000001)))**

**if den == 0:**

**H = 0**

**elif B[i, j] <= G[i, j]:**

**H = theta**

**else:**

**H = 2 \* 3.14169265 - theta**

**min\_RGB = min(min(B[i, j], G[i, j]), R[i, j])**

**sum = B[i, j] + G[i, j] + R[i, j]**

**if sum == 0:**

**S = 0**

**else:**

**S = 1 - 3 \* min\_RGB / sum**

**H = H / (2 \* 3.14159265)**

**I = sum / 3.0**

**# 輸出HSI圖像，擴充到255以方便顯示，一般H分量在[0,2pi]之間，S和I在[0,1]之間**

**hsi\_lwpImg[i, j, 0] = H \* 255**

**hsi\_lwpImg[i, j, 1] = S \* 255**

**hsi\_lwpImg[i, j, 2] = I \* 255**

**return hsi\_lwpImg**

**def hsitorgb(hsi\_img):**

**h = int(hsi\_img.shape[0])**

**w = int(hsi\_img.shape[1])**

**H, S, I = cv2.split(hsi\_img)**

**H = H / 255.0**

**S = S / 255.0**

**I = I / 255.0**

**bgr\_img = hsi\_img.copy()**

**B, G, R = cv2.split(bgr\_img)**

**for i in range(h):**

**for j in range(w):**

**if S[i, j] < 1e-6:**

**R = I[i, j]**

**G = I[i, j]**

**B = I[i, j]**

**else:**

**H[i, j] \*= 360**

**if H[i, j] > 0 and H[i, j] <= 120:**

**B = I[i, j] \* (1 - S[i, j])**

**R = I[i, j] \* (1 + (S[i, j] \* math.cos(H[i, j]\*math.pi/180)) / math.cos((60 - H[i, j])\*math.pi/180))**

**G = 3 \* I[i, j] - (R + B)**

**elif H[i, j] > 120 and H[i, j] <= 240:**

**H[i, j] = H[i, j] - 120**

**R = I[i, j] \* (1 - S[i, j])**

**G = I[i, j] \* (1 + (S[i, j] \* math.cos(H[i, j]\*math.pi/180)) / math.cos((60 - H[i, j])\*math.pi/180))**

**B = 3 \* I[i, j] - (R + G)**

**elif H[i, j] > 240 and H[i, j] <= 360:**

**H[i, j] = H[i, j] - 240**

**G = I[i, j] \* (1 - S[i, j])**

**B = I[i, j] \* (1 + (S[i, j] \* math.cos(H[i, j]\*math.pi/180)) / math.cos((60 - H[i, j])\*math.pi/180))**

**R = 3 \* I[i, j] - (G + B)**

**bgr\_img[i, j, 0] = B \* 255**

**bgr\_img[i, j, 1] = G \* 255**

**bgr\_img[i, j, 2] = R \* 255**

**return bgr\_img**

**Comments：**

**這次得作業算是相當的困難,主要是難在要將rgb轉成hsi以及hsi轉成rgb。首先因為這兩種轉換opencv都沒有套件,必須要自己寫,網路上看到大部分也都只有rgb轉成hsi,hsi轉成rgb的資訊卻很少,大部分都是在對hsv轉換。Histogram equalization 就比照之前做的作業即可。轉換時也要很注意資料型態以及0在分母的問題,花了蠻久時間的。**