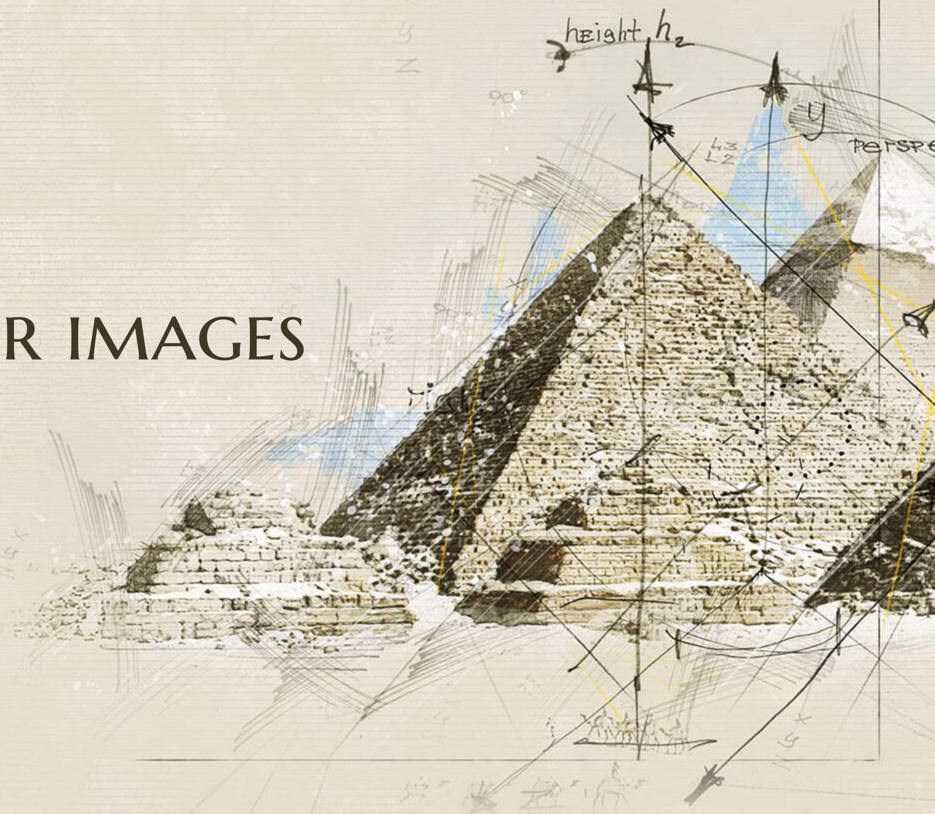
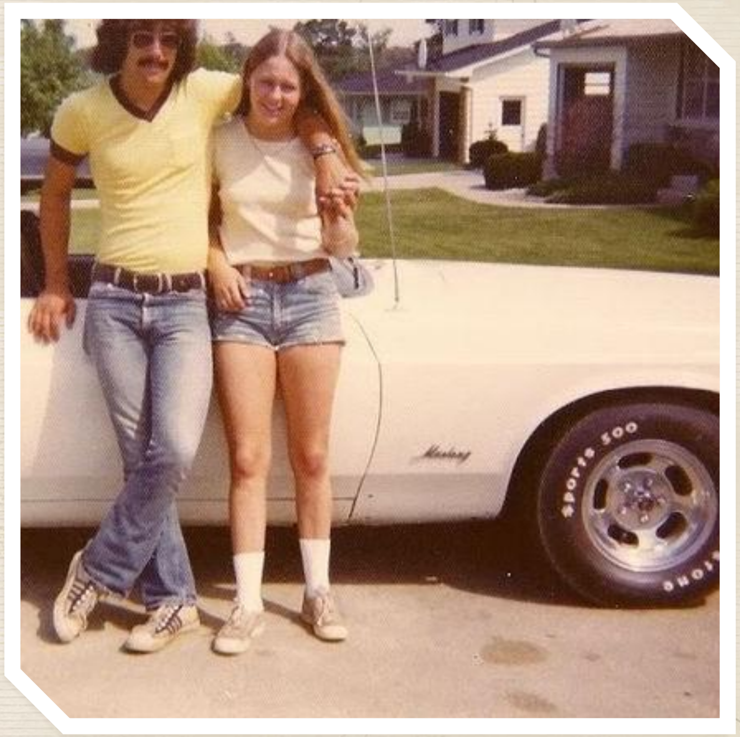
# Activity 6
## Enhancement of color images



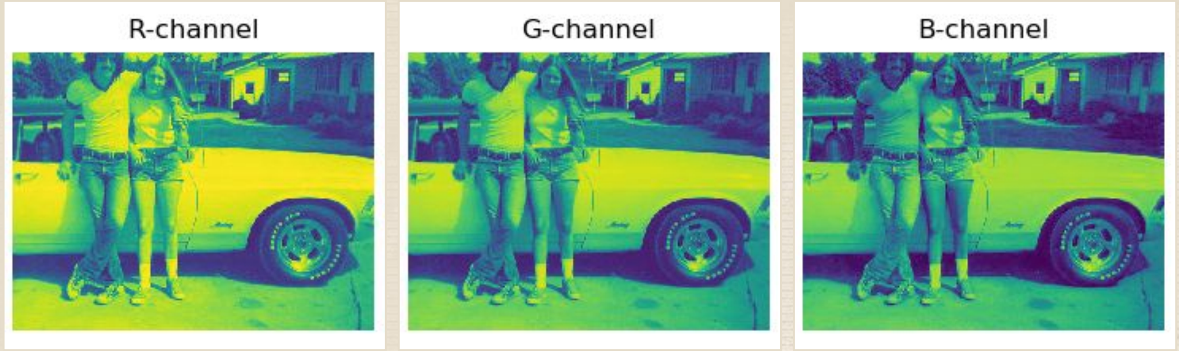
## Original image

To be done:

- contrast stretching in RGB
- gray world algorithm
- white patch algorithm



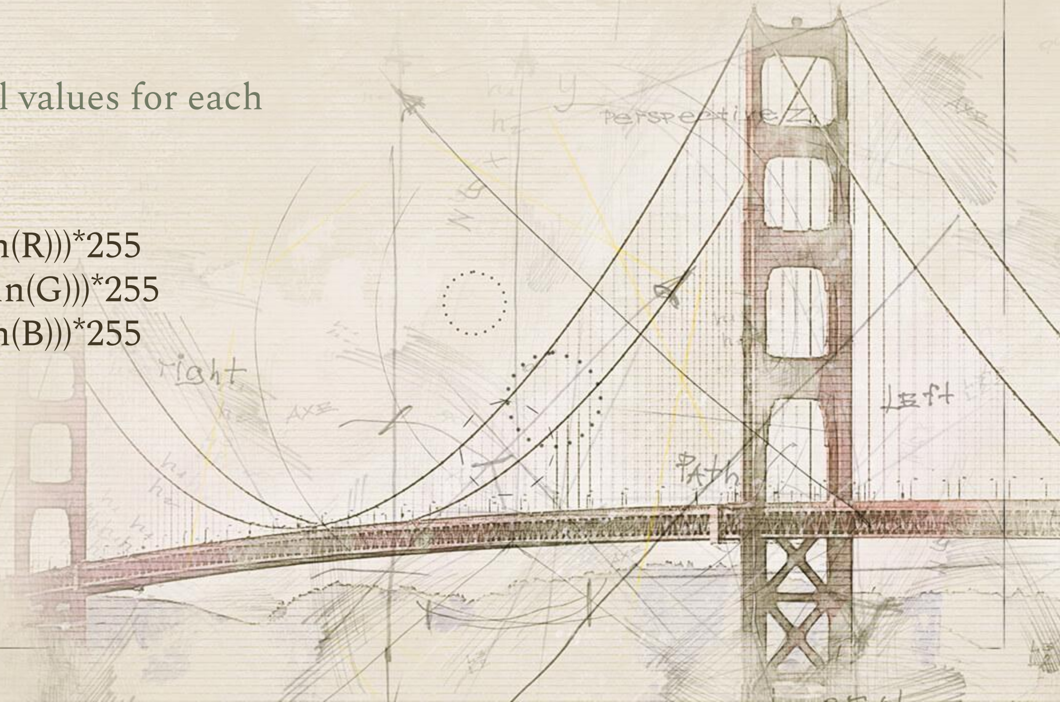
## 1. Contrast stretching

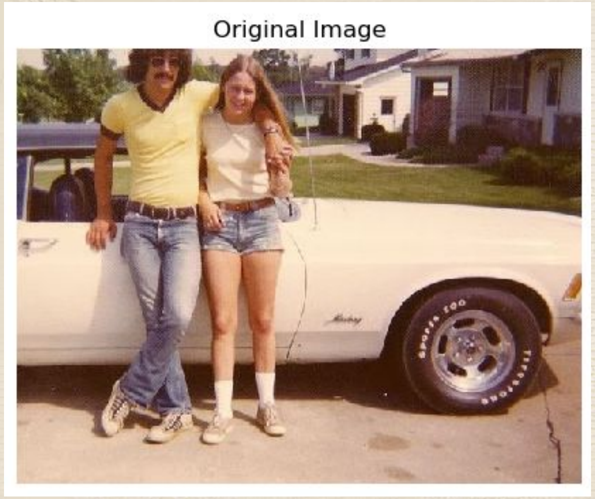


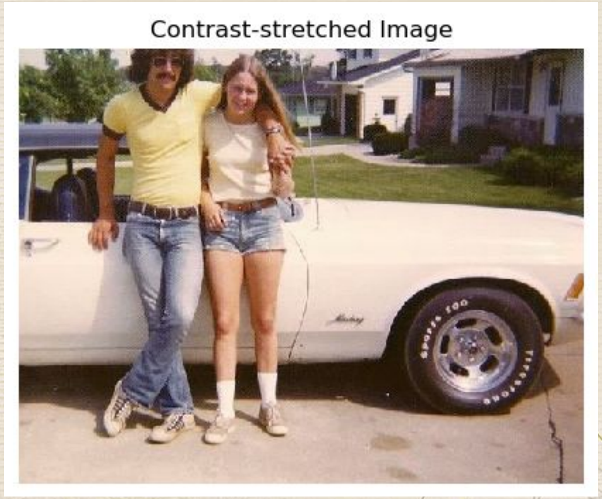
Using the split() function from OpenCV, the separate R,G,B color channels were obtained. For each channel, contrast stretching (also called ‘normalization’) was then done to “stretch” the pixel values to the minimum (0) and maximum values (255).

## Code snippet:

```
#stretching the minimum and maximum pixel values for each color channel
R,G,B = cv2.split(contrast)
Rstretched = ((R-np.min(R))/(np.max(R)-np.min(R)))*255
Gstretched = ((G-np.min(G))/(np.max(G)-np.min(G)))*255
Bstretched = ((B-np.min(B))/(np.max(B)-np.min(B)))*255
```

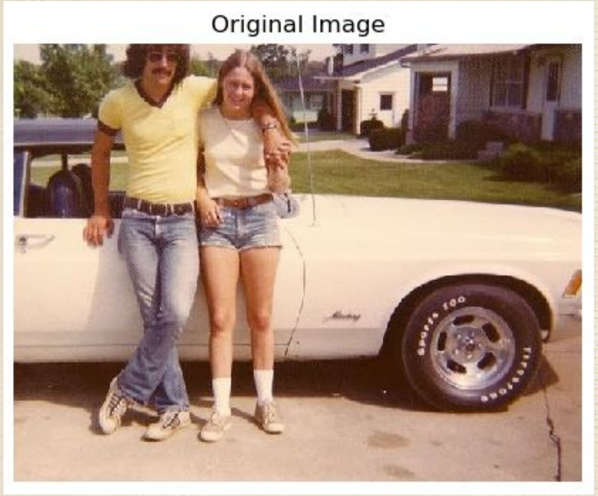


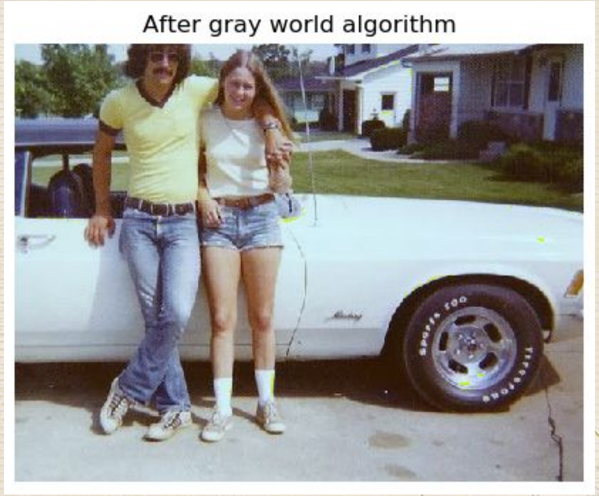




Comparing the two images, it can be seen that the manipulated image was indeed enhanced-- the contrasts in the image are improved. This change can be seen more clearly in the brighter-colored parts of the image.

## 2. Gray world algorithm

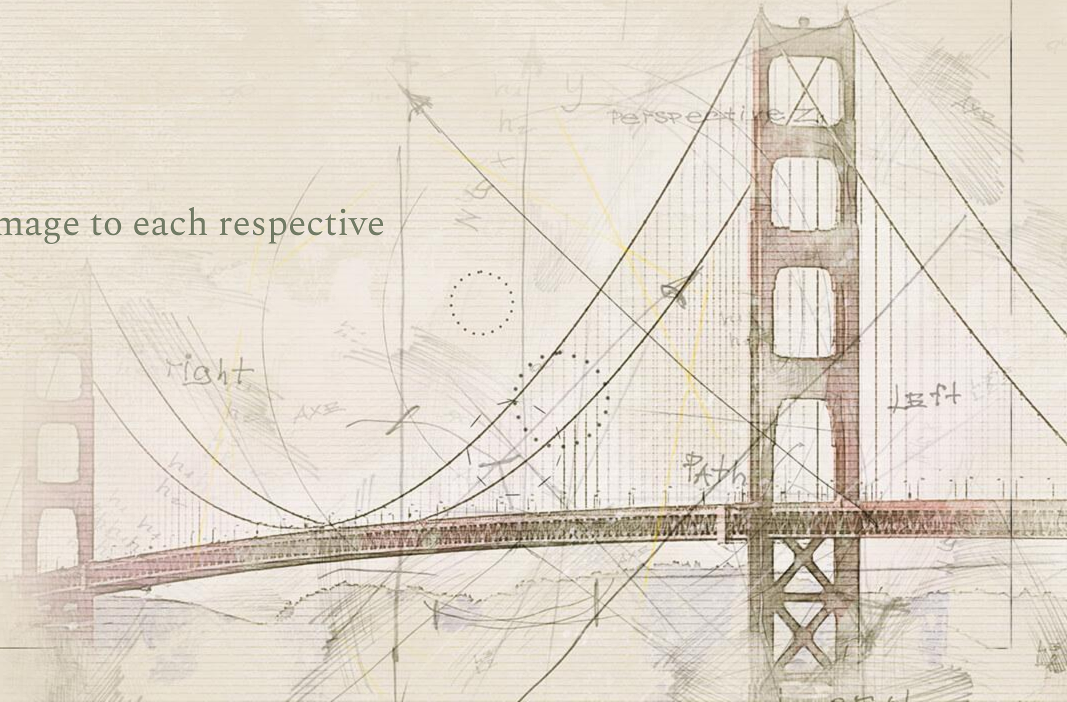




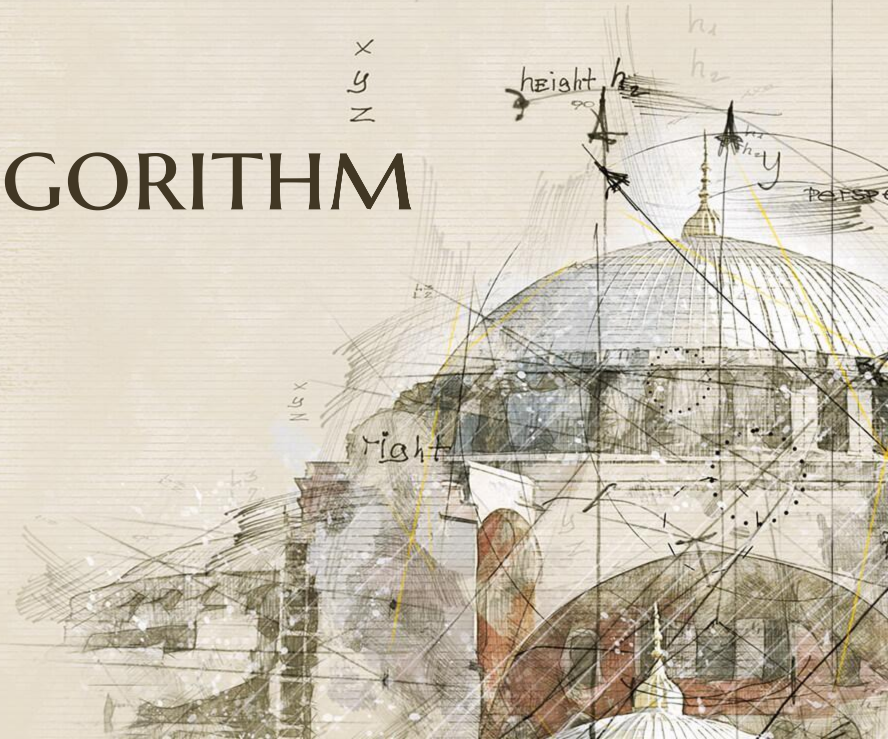
The gray world algorithm does a decent job here. It did a significant change on the orangish color of the original image. However, the final manipulated image developed a bluish tint, which is most noticeable on the white parts of the image.

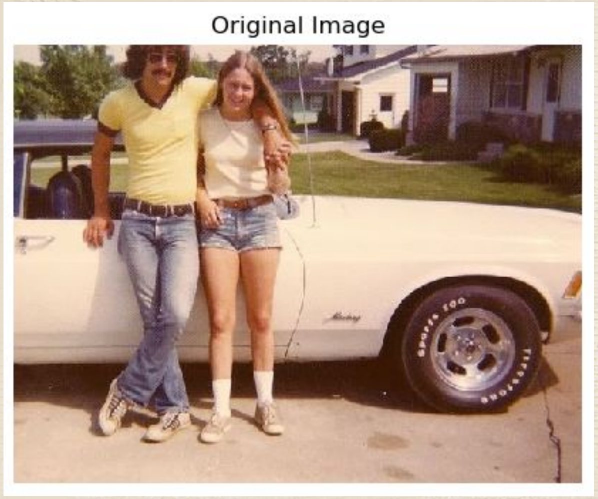
## Code snippet:

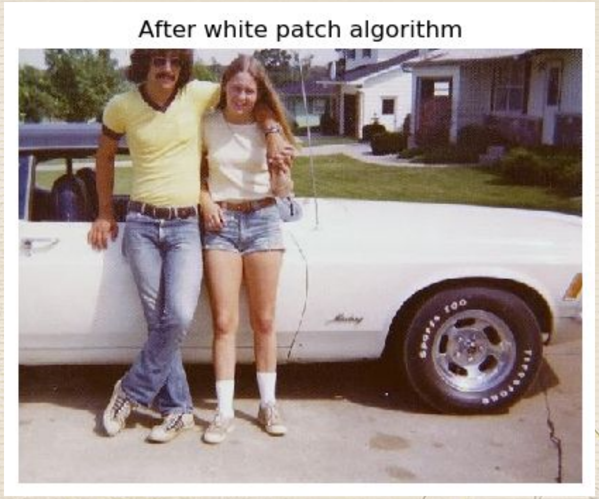
```
#getting the average for each color channel
R_ave = (np.sum(R)/(R.shape[0]*R.shape[1]))
G_ave = (np.sum(G)/(G.shape[0]*G.shape[1]))
B_ave = (np.sum(B)/(B.shape[0]*B.shape[1]))
#dividing each color channel of the original image to each respective mean color channel
R_gw = (R/R_ave)
G_gw = (G/G_ave)
B_gw = (B/B_ave)
```



## 3. White patch algorithm



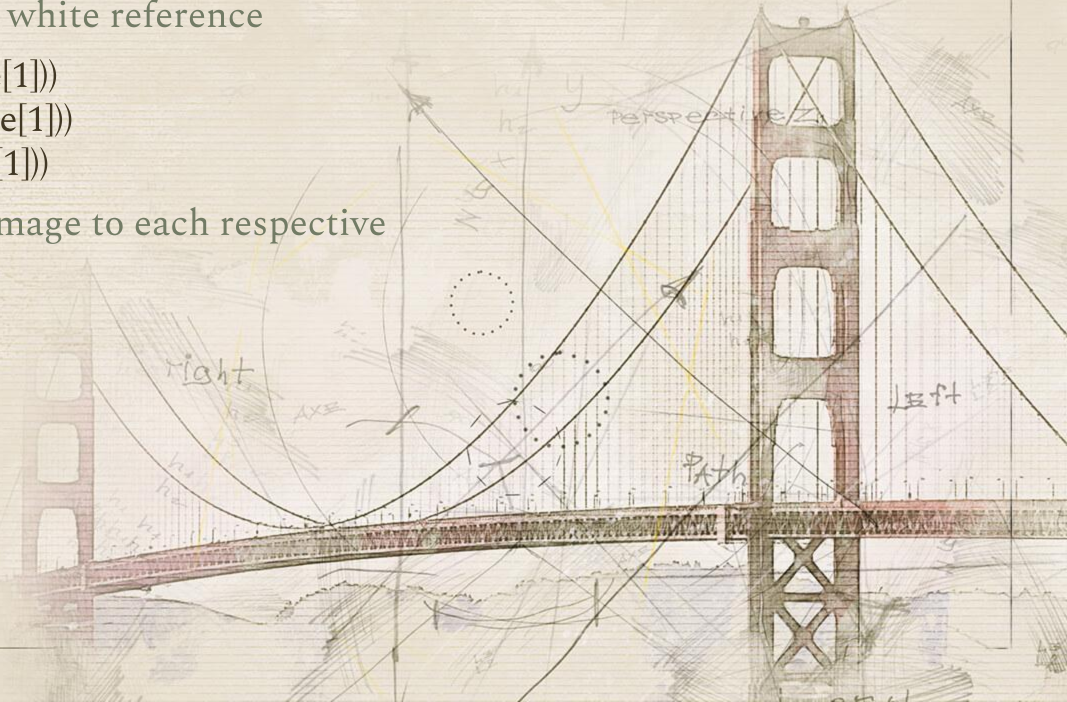




I used the white car as my white reference in doing the algorithm, as it is the lightest patch on the image. It can be seen on the treated image that the original orangish tint was removed. Just by seeing the whites in the car from before and after the image processing, it can already be concluded that this technique is a great white balancing algorithm.

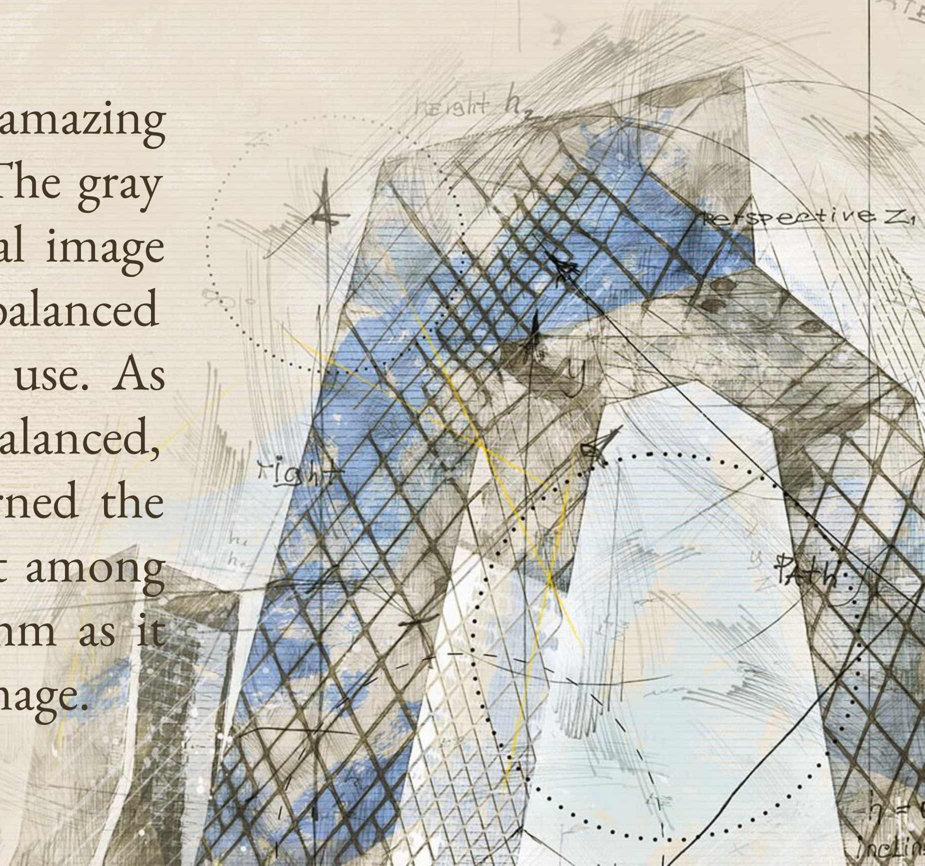
## Code snippet:

```
#white averages for each color channel of the white reference

Rw_ave = (np.sum(Rw)/(Rw.shape[0]*Rw.shape[1]))
Gw_ave = (np.sum(Gw)/(Gw.shape[0]*Gw.shape[1]))
Bw_ave = (np.sum(Bw)/(B1.shape[0]*Bw.shape[1]))

#dividing each color channel of the original image to each respective
white averages

R_wp = (R/Rw_ave)
G_wp = (G/Gw_ave)
B_wp = (B/Bw_ave)
```



## Summary

All the white balancing algorithms did an amazing job in enhancing the color faded image. The gray world algorithm depended on the original image used. I observed that using a well color-balanced photo makes this algorithm sufficient to use. As the photo I used is not that color-balanced, treating it with gray world algorithm turned the photo to one with a bluish tint. The best among the algorithms is the white patch algorithm as it removed the orangish tint of my original image.



# Self-evaluation

| | |
|---|---|
| Technical correctness | 5 |
| Quality of presentation | 4 |
| Initiative | 2 |
| **Total** | **11** |