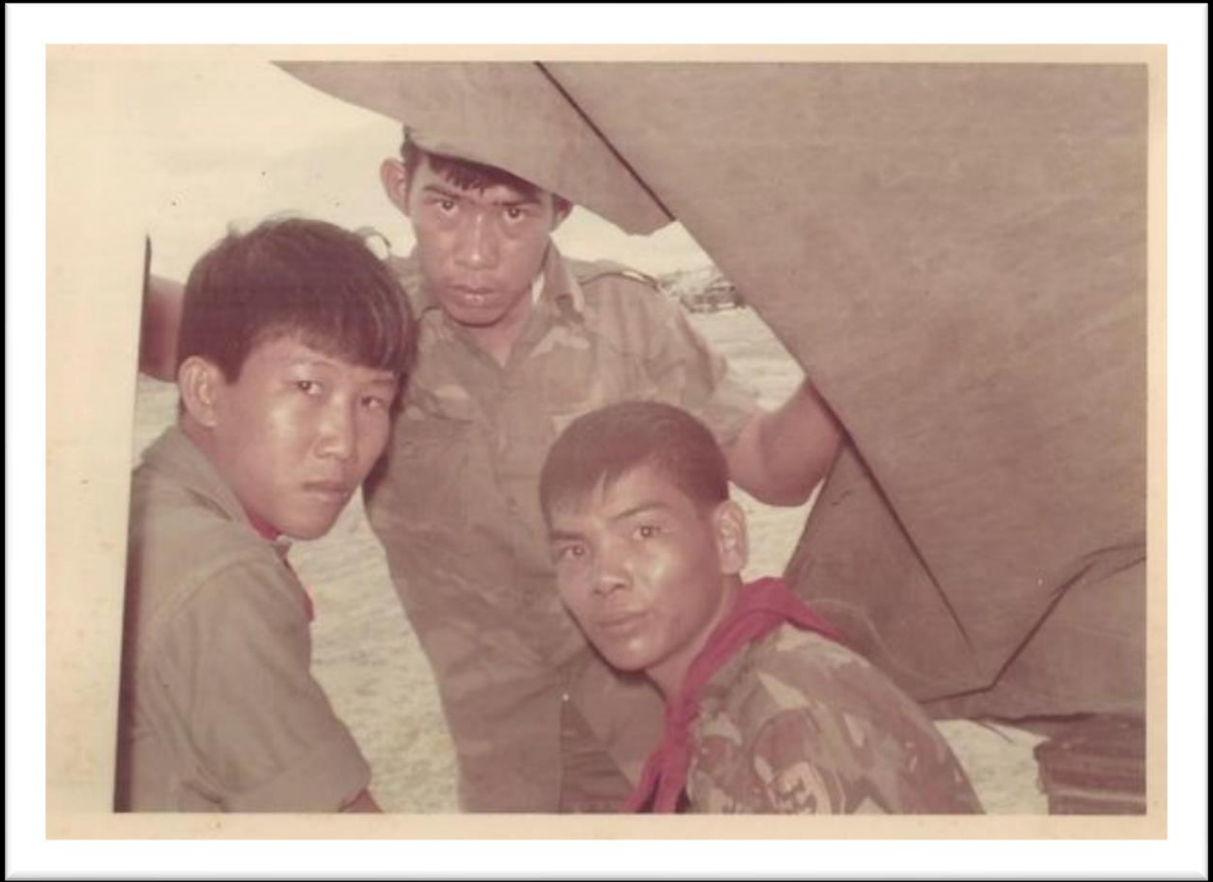
ENHANCEMENT OF COLOR IMAGES

ACTIVITY 6

Krishna Lyn Delima
2014-64503

Original Image



Photos from the past usually degrade overtime due to exposure to UV light that severs chemical bonds present in the picture dye. [1] This picture was taken in Vietnam, 1968.

Image taken from: https://www.etsy.com/BettywasaBombshell/listing/576015884/vietnam-1968-base-called-concord-ranger?utm_source=Pinterest&utm_medium=ListingManager&utm_campaign=Share&utm_term=so.lmsm&share_time=1517176306343

Restoration using various methods in White Balancing:

- Contrast Stretching
- Gray World Algorithm
- White Patch Algorithm

Done using:

- Jupyter Notebook (Python)
  - Packages:

    matplotlib.pyplot

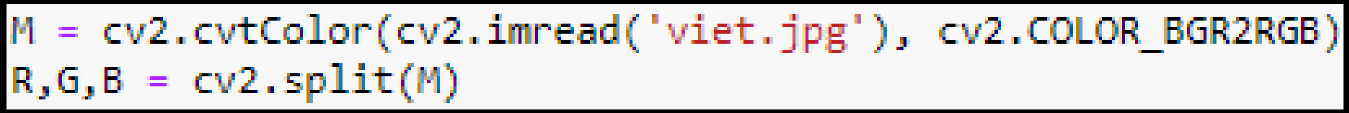
    numpy

    cv2
- Photos taken from Google Images and Pinterest

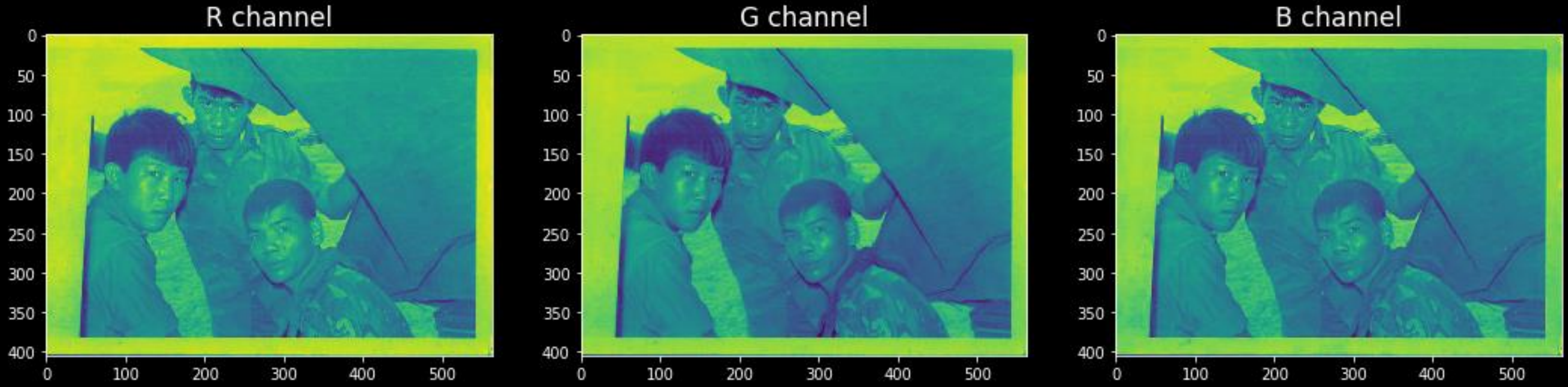
Contrast Stretching

1

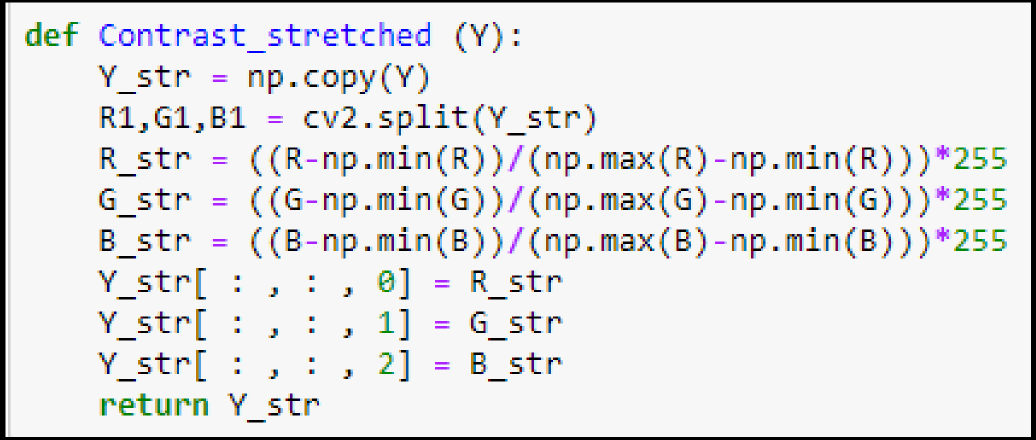
Step 1 Splitting RGB Channels



```
M = cv2.cvtColor(cv2.imread('viet.jpg'), cv2.COLOR_BGR2RGB)
R,G,B = cv2.split(M)
```



Step 2 Contrast Stretching

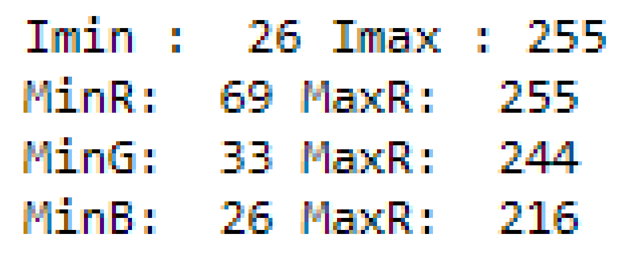


```
def Contrast_stretched (Y):
    Y_str = np.copy(Y)
    R1,G1,B1 = cv2.split(Y_str)
    R_str = ((R-np.min(R))/(np.max(R)-np.min(R)))*255
    G_str = ((G-np.min(G))/(np.max(G)-np.min(G)))*255
    B_str = ((B-np.min(B))/(np.max(B)-np.min(B)))*255
    Y_str[ : , : , 0] = R_str
    Y_str[ : , : , 1] = G_str
    Y_str[ : , : , 2] = B_str
    return Y_str
```

- Normalizing pixel values per RGB channel and by multiplying it to 255 (since it’s 8-bit) did the trick!

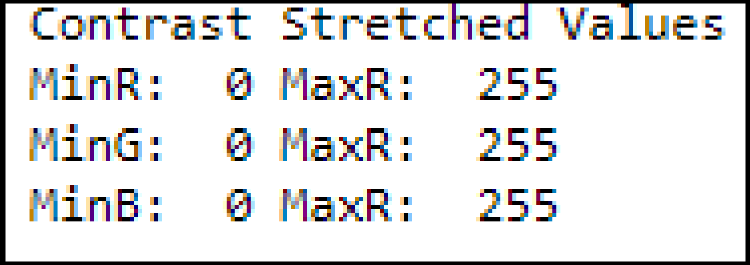
Step 2 Contrast Stretching

Original values



```
Imin :   26 Imax : 255
MinR:   69 MaxR:   255
MinG:   33 MaxR:   244
MinB:   26 MaxR:   216
```

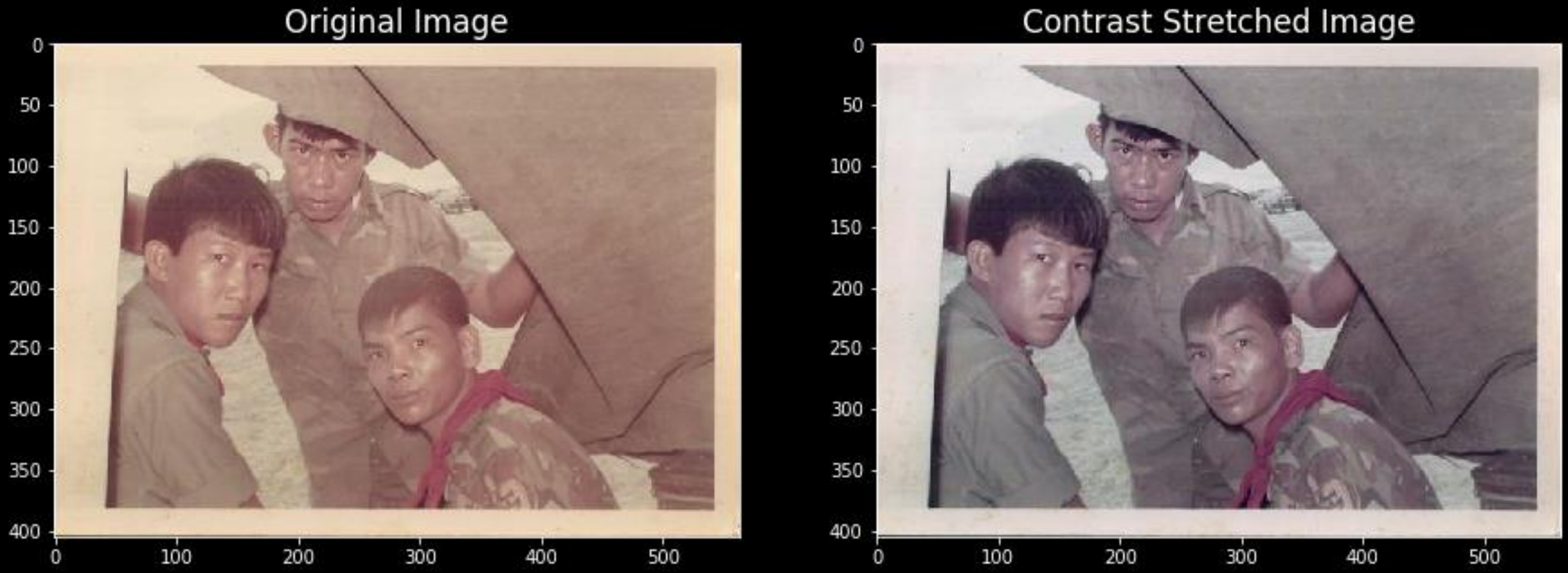
Resulting values



```
Contrast Stretched Values
MinR:   0 MaxR:   255
MinG:   0 MaxR:   255
MinB:   0 MaxR:   255
```

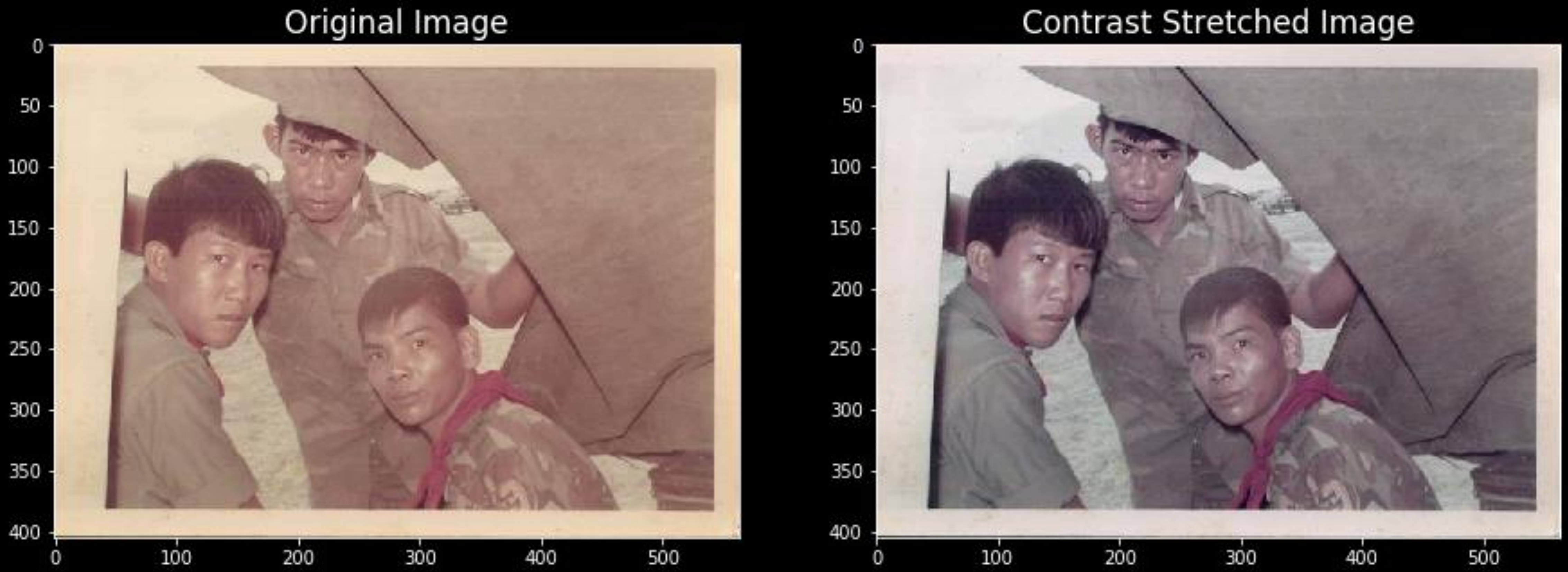
- Contrast stretching is used for low contrast photos
  - original maximum and minimum values were set to 0 and 1 after normalizing
  - results to a higher contrast image (darker on dark areas and brighter on bright areas)

Step 3 Results



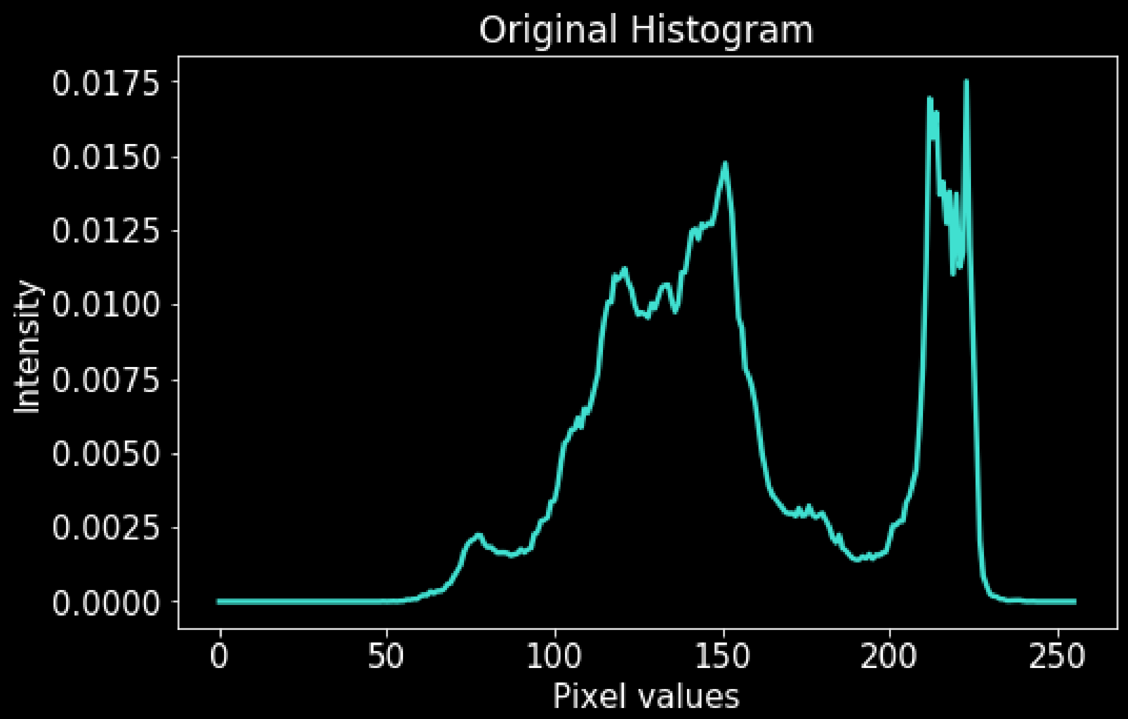
Resulted to an image with slightly higher contrast considering the original Imax (max pixel value)

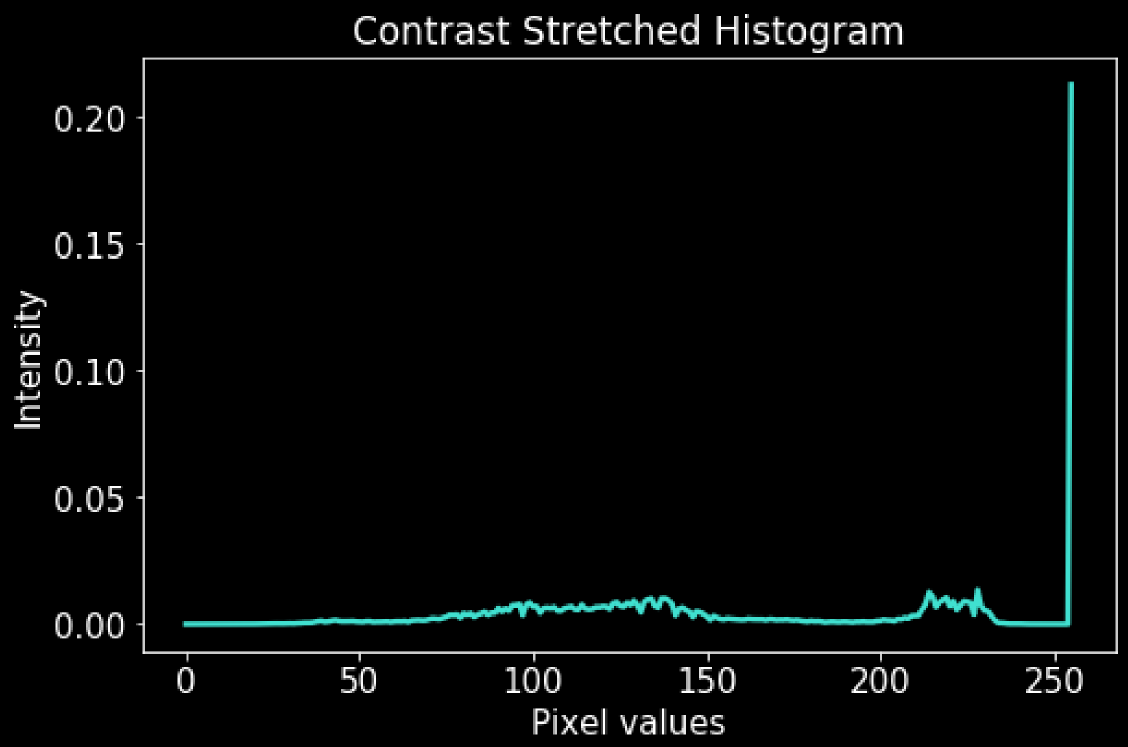
Step 3 Results



- Contrast change is quite obvious
- Actual colors have emerged; the green in their uniform seems more distinguishable now, and the red scarf is a lot more red than before.

Step 3 Results: Histogram





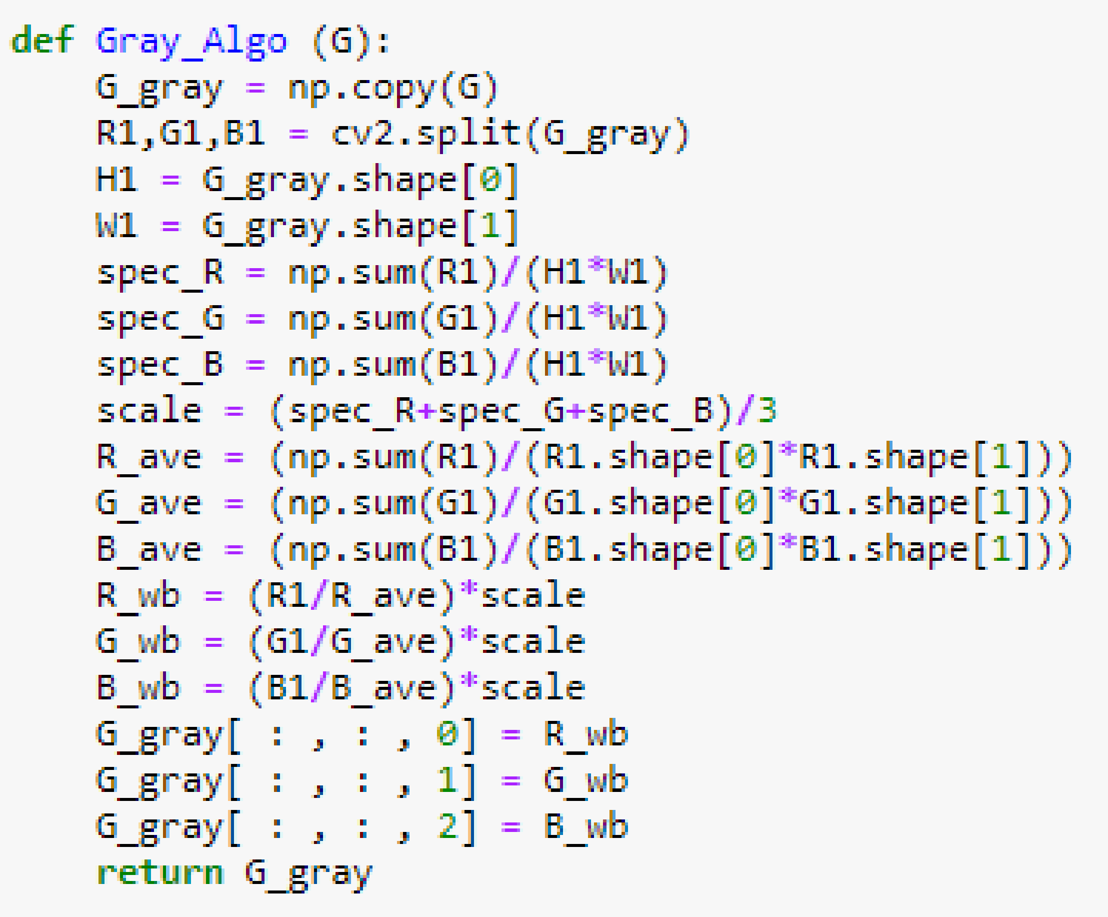
- The shift in contrast is evident in the original and contrast stretched histogram of the image

Gray World Algorithm

2

Step 1 Gray World Algorithm

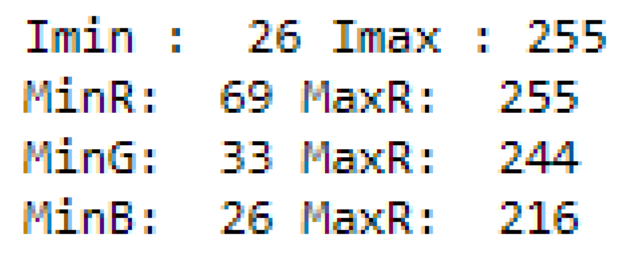
- Averaging each RGB channel enables to approximate these colors to gray
- A balancing constant was formulated
- These balancing constants were then scaled



```
def Gray_Algo (G):
    G_gray = np.copy(G)
    R1,G1,B1 = cv2.split(G_gray)
    H1 = G_gray.shape[0]
    W1 = G_gray.shape[1]
    spec_R = np.sum(R1)/(H1*W1)
    spec_G = np.sum(G1)/(H1*W1)
    spec_B = np.sum(B1)/(H1*W1)
    scale = (spec_R+spec_G+spec_B)/3
    R_ave = (np.sum(R1)/(R1.shape[0]*R1.shape[1]))
    G_ave = (np.sum(G1)/(G1.shape[0]*G1.shape[1]))
    B_ave = (np.sum(B1)/(B1.shape[0]*B1.shape[1]))
    R_wb = (R1/R_ave)*scale
    G_wb = (G1/G_ave)*scale
    B_wb = (B1/B_ave)*scale
    G_gray[ : , : , 0] = R_wb
    G_gray[ : , : , 1] = G_wb
    G_gray[ : , : , 2] = B_wb
    return G_gray
```

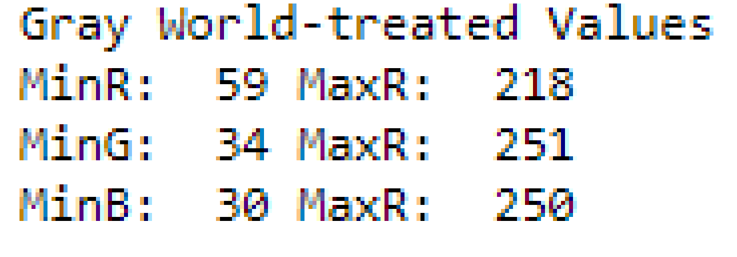
Step 2 Gray World Algorithm

Original values



```
Imin :   26 Imax : 255
MinR:   69 MaxR:   255
MinG:   33 MaxR:   244
MinB:   26 MaxR:   216
```

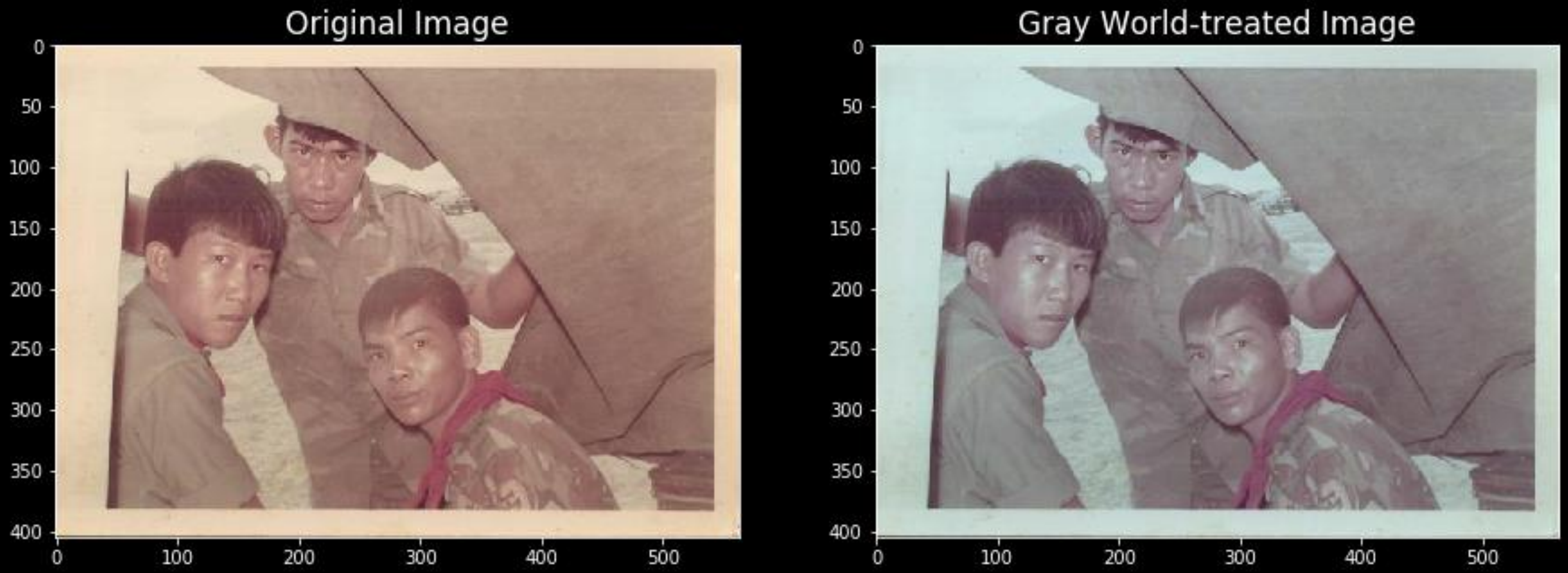
Resulting values



```
Gray World-treated Values
MinR:   59 MaxR:   218
MinG:   34 MaxR:   251
MinB:   30 MaxR:   250
```

- In **Gray World Algorithm,** gray becomes the average of all colors
  - There's an apparent shift of minimum and maximum pixel values from the Original to the Gray World- treated values.

Step 3 Results

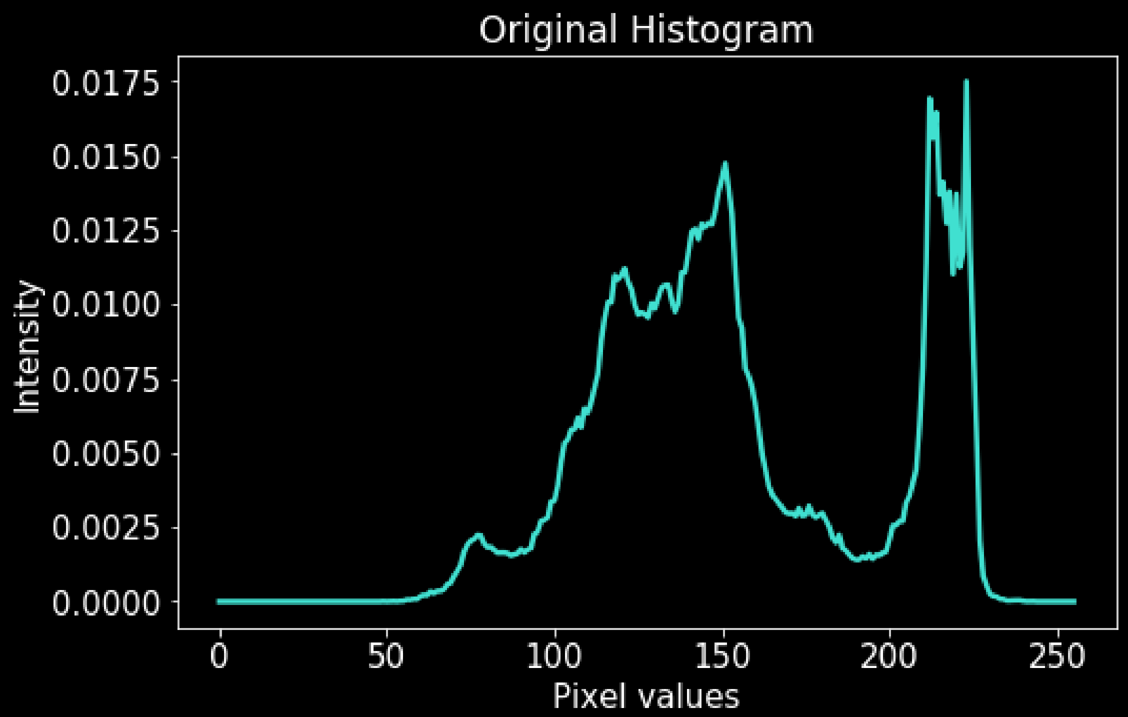


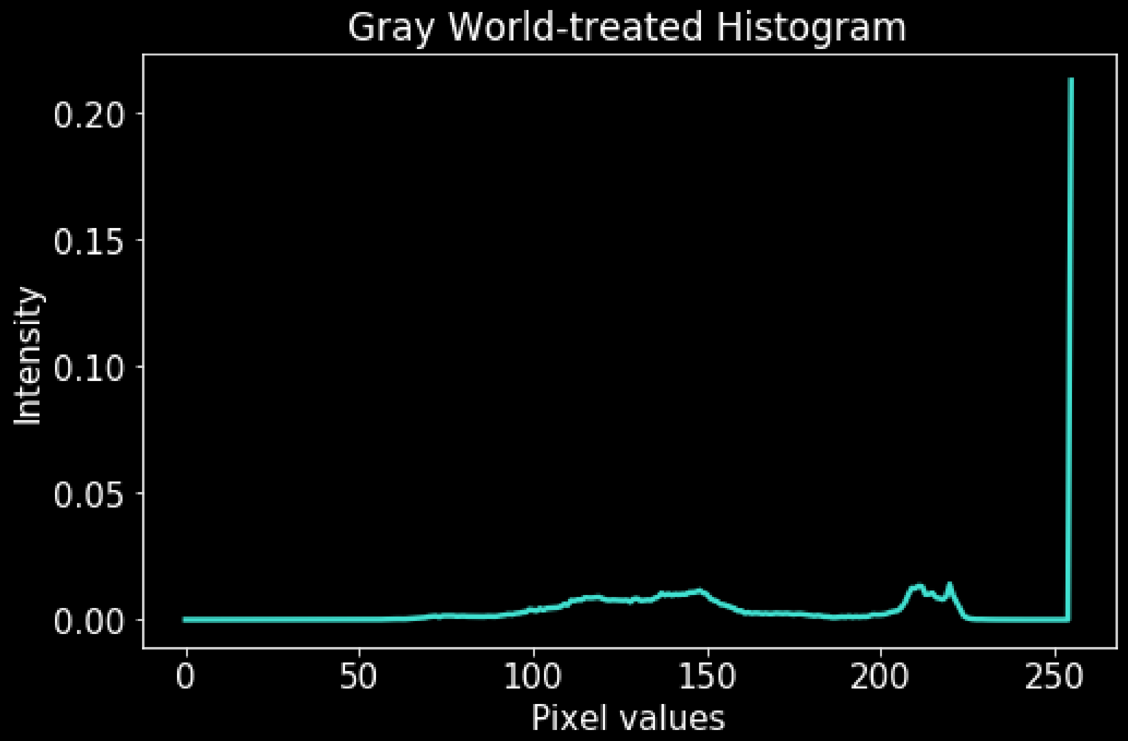
Original Image

Gray World-treated Image

- Actual colors have emerged slightly but the whole image has a bluish tinge.

Step 3 Results: Histogram



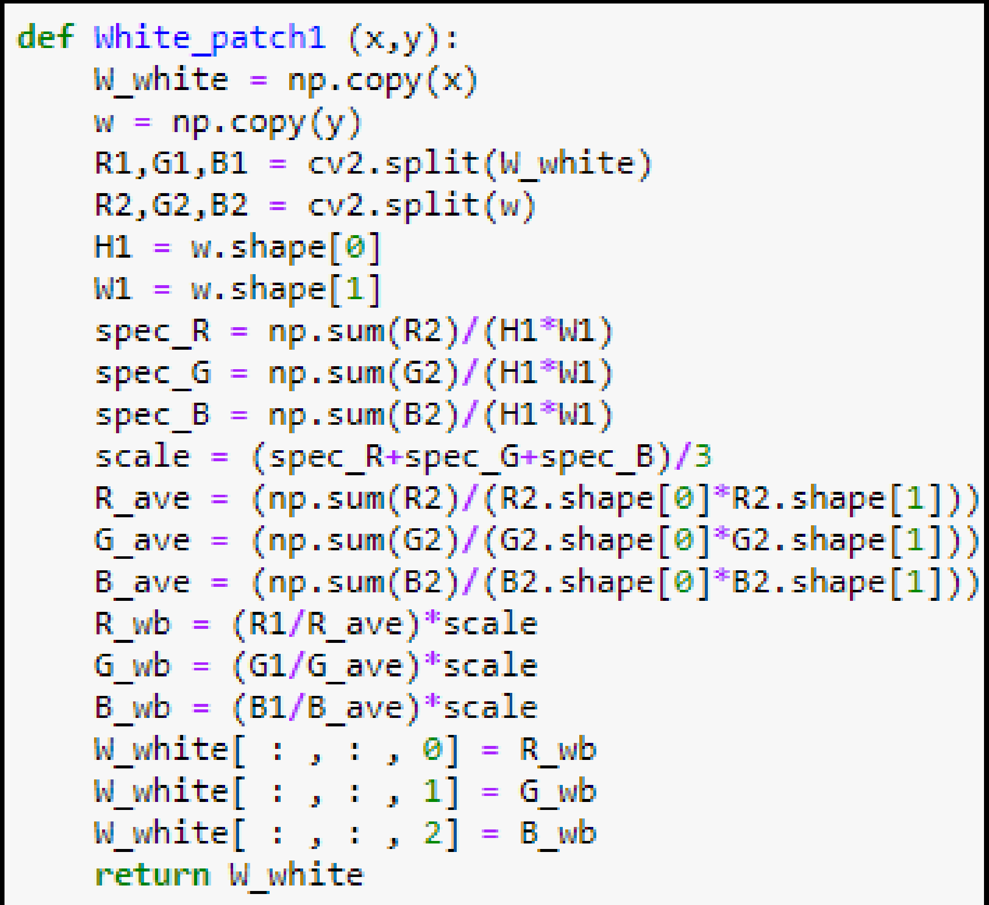


- The shift in contrast is evident

White Patch Algorithm

3

Step 1 White Patch Algorithm

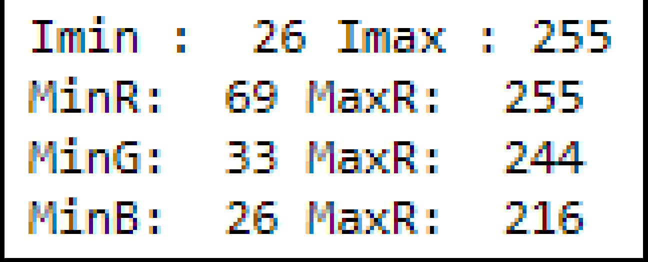


```
def White_patch1 (x,y):
    W_white = np.copy(x)
    w = np.copy(y)
    R1,G1,B1 = cv2.split(W_white)
    R2,G2,B2 = cv2.split(w)
    H1 = w.shape[0]
    W1 = w.shape[1]
    spec_R = np.sum(R2)/(H1*W1)
    spec_G = np.sum(G2)/(H1*W1)
    spec_B = np.sum(B2)/(H1*W1)
    scale = (spec_R+spec_G+spec_B)/3
    R_ave = (np.sum(R2)/(R2.shape[0]*R2.shape[1]))
    G_ave = (np.sum(G2)/(G2.shape[0]*G2.shape[1]))
    B_ave = (np.sum(B2)/(B2.shape[0]*B2.shape[1]))
    R_wb = (R1/R_ave)*scale
    G_wb = (G1/G_ave)*scale
    B_wb = (B1/B_ave)*scale
    W_white[ : , : , 0] = R_wb
    W_white[ : , : , 1] = G_wb
    W_white[ : , : , 2] = B_wb
    return W_white
```

- This is similar to the Gray World Algorithm but the one used for averaging is a certain patch (or region) on the image that’s white (or supposed to be white).

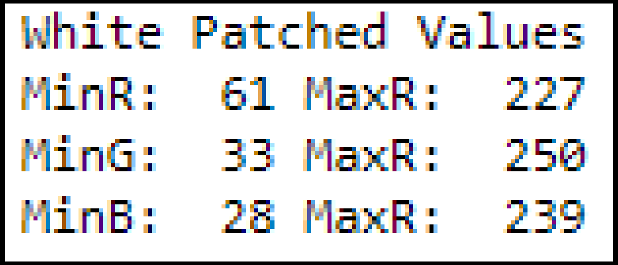
Step 2 White Patch Algorithm

Original values



```
Imin :   26 Imax : 255
MinR:  69 MaxR:   255
MinG:  33 MaxR:   244
MinB:  26 MaxR:   216
```

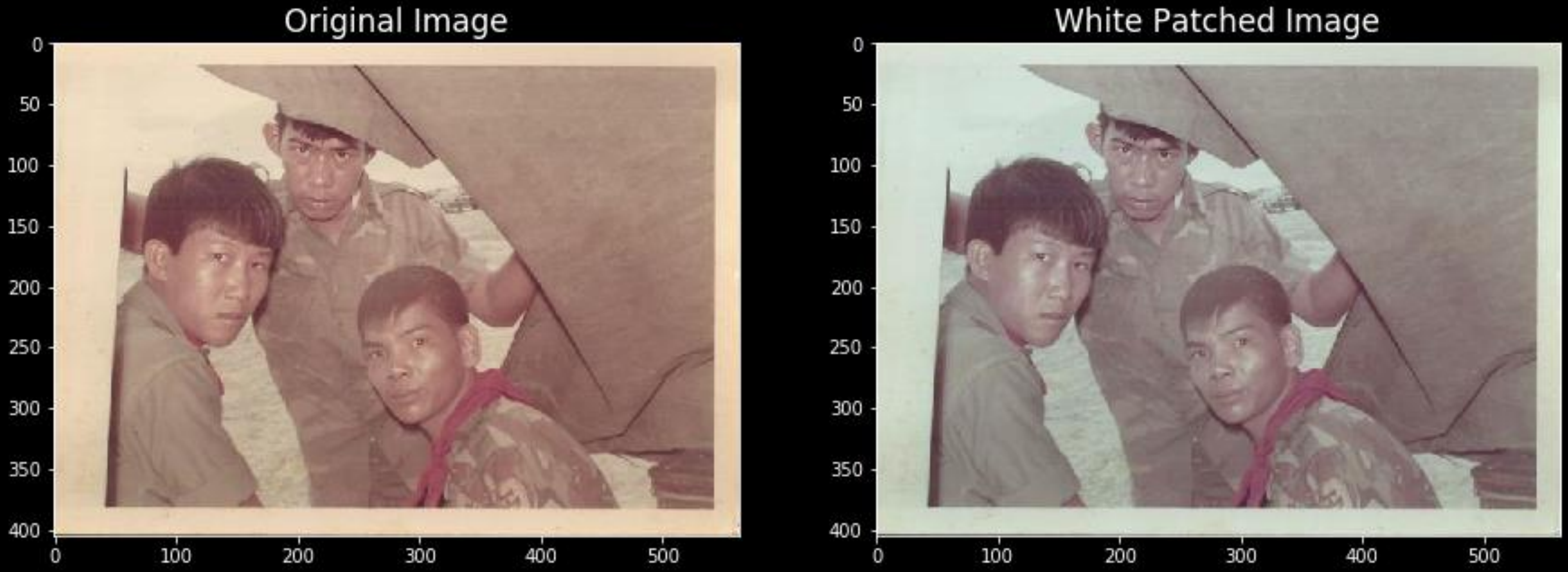
Resulting values



```
White Patched Values
MinR:   61 MaxR:   227
MinG:   33 MaxR:   250
MinB:   28 MaxR:   239
```

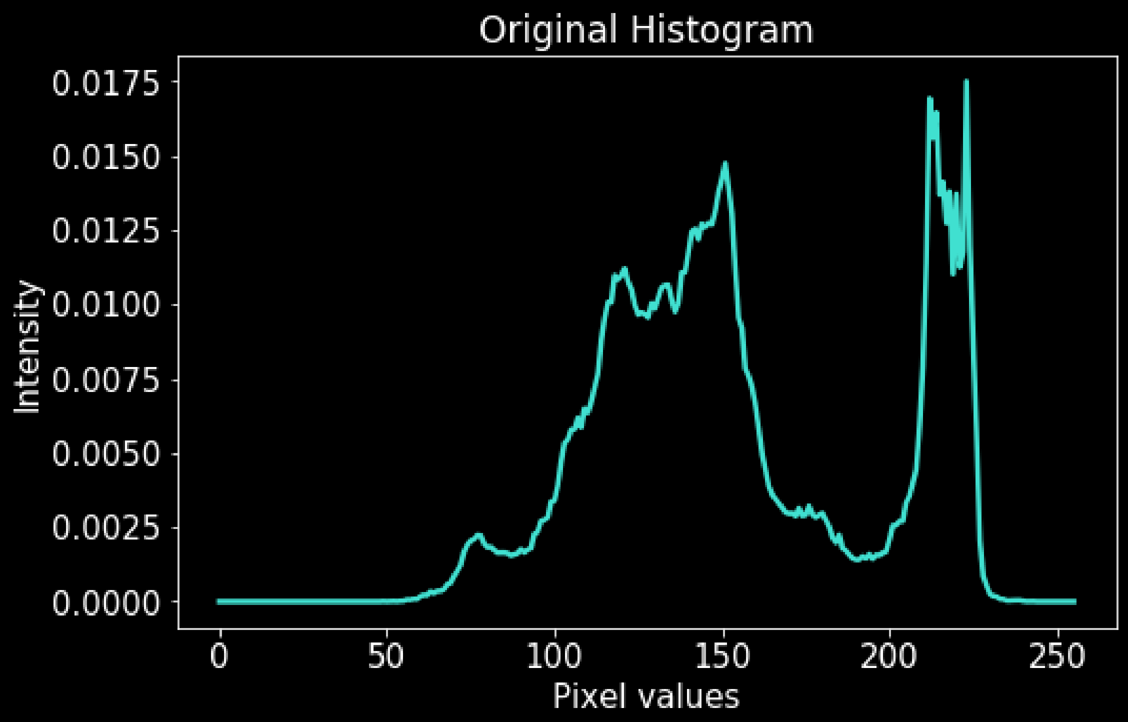
- In **White Patch Algorithm,** a known white object present in the image is used as the basis for altering RGB values
  - Minimum values haven't deviated much compared to the deviation in their maximum values → a shift to white

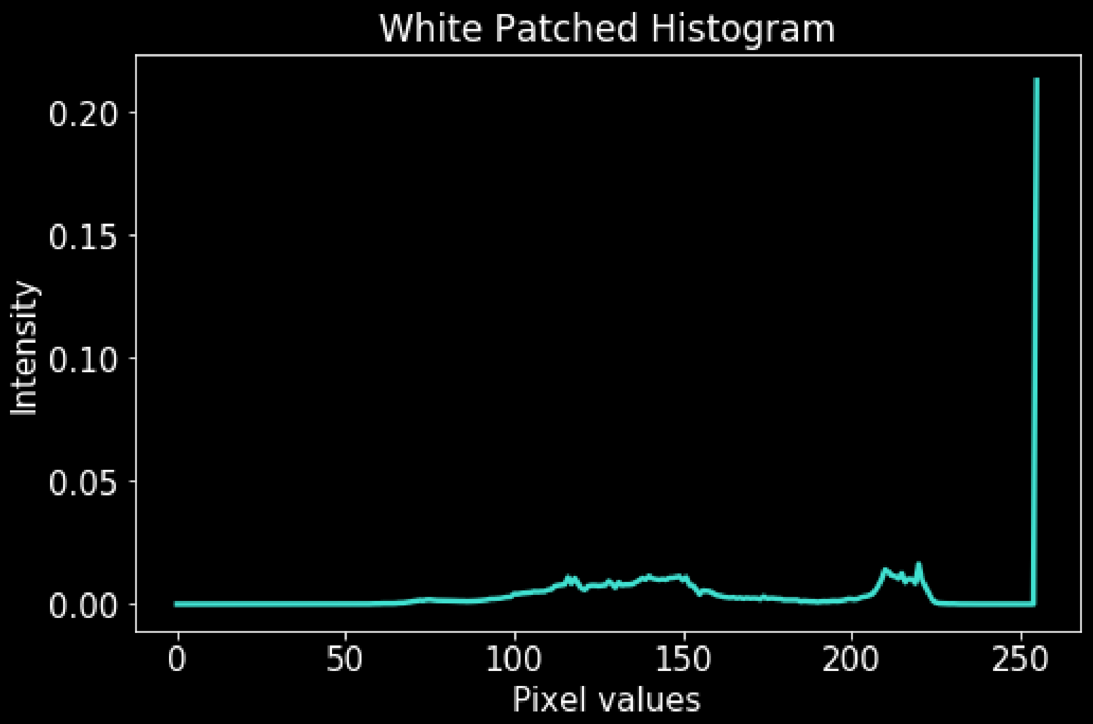
Step 3 Results



- There’s an apparent shift to white due to the whitish tinge present. Due to this, there’s an increase in brightness.

Step 3 Results: Histogram



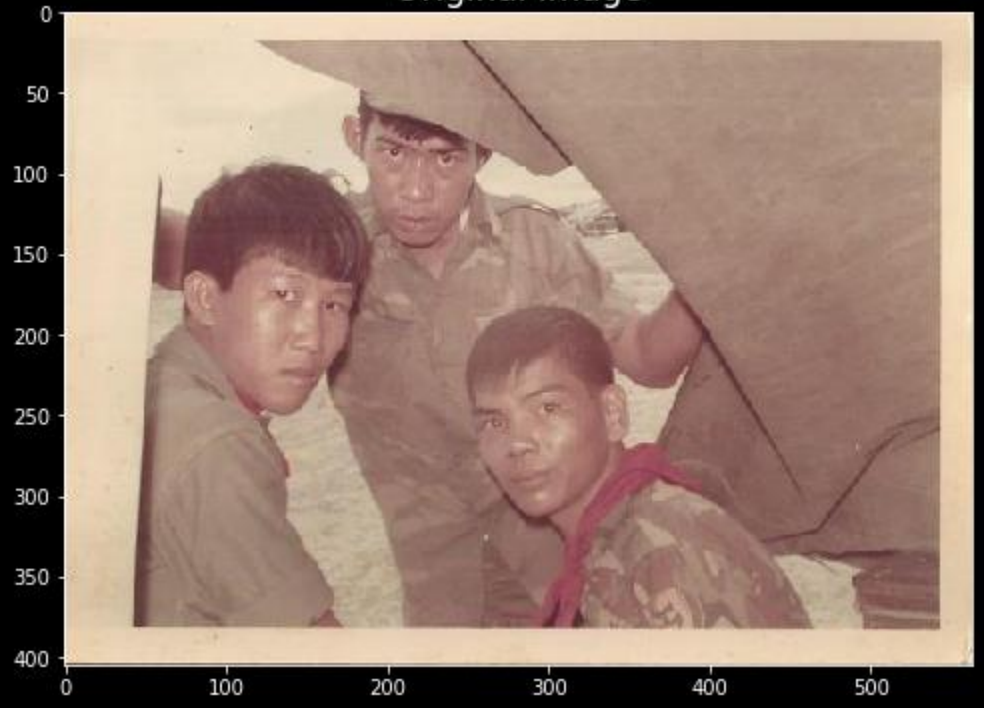


- The shift in contrast is evident in this one too.

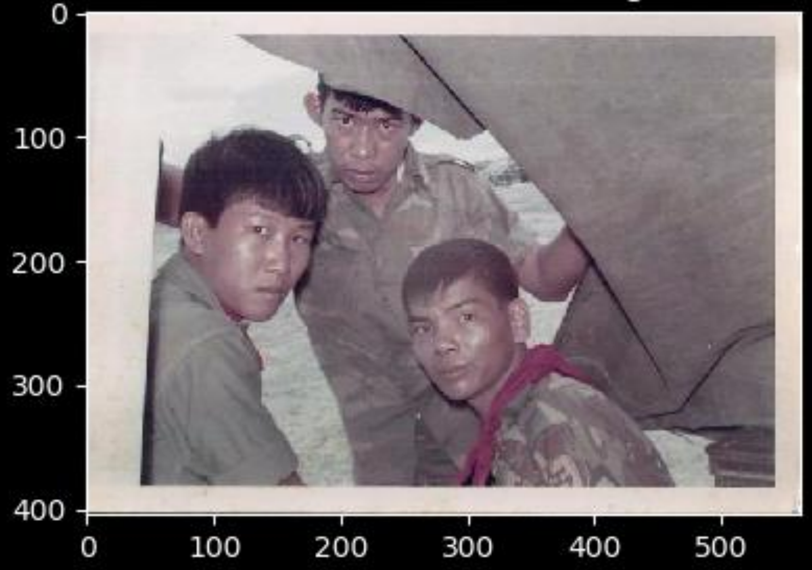
* Comparison

- All methods of restoration has depleted the orange tinge from the original image
- Sharpest Image : Contrast Stretch
- Brightest Image : White Patch Algorithm

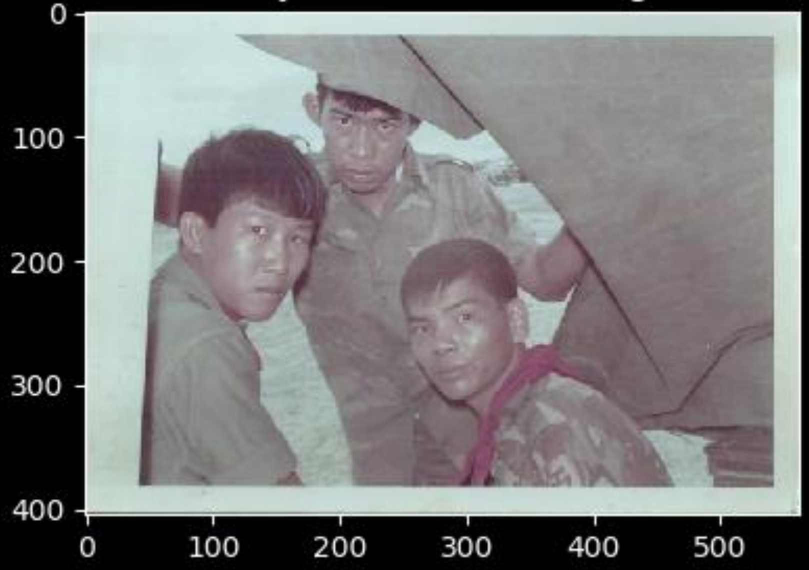
Original Image



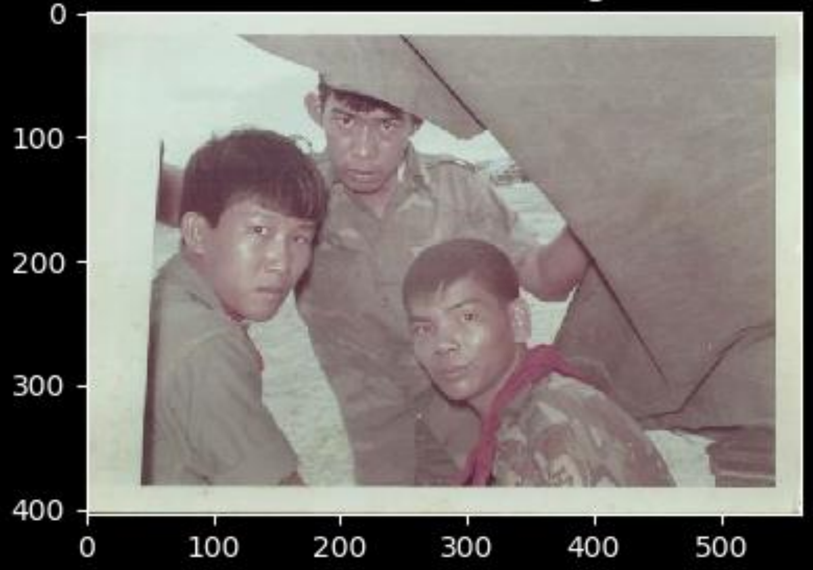
Contrast Stretched Image



Gray World-treated Image



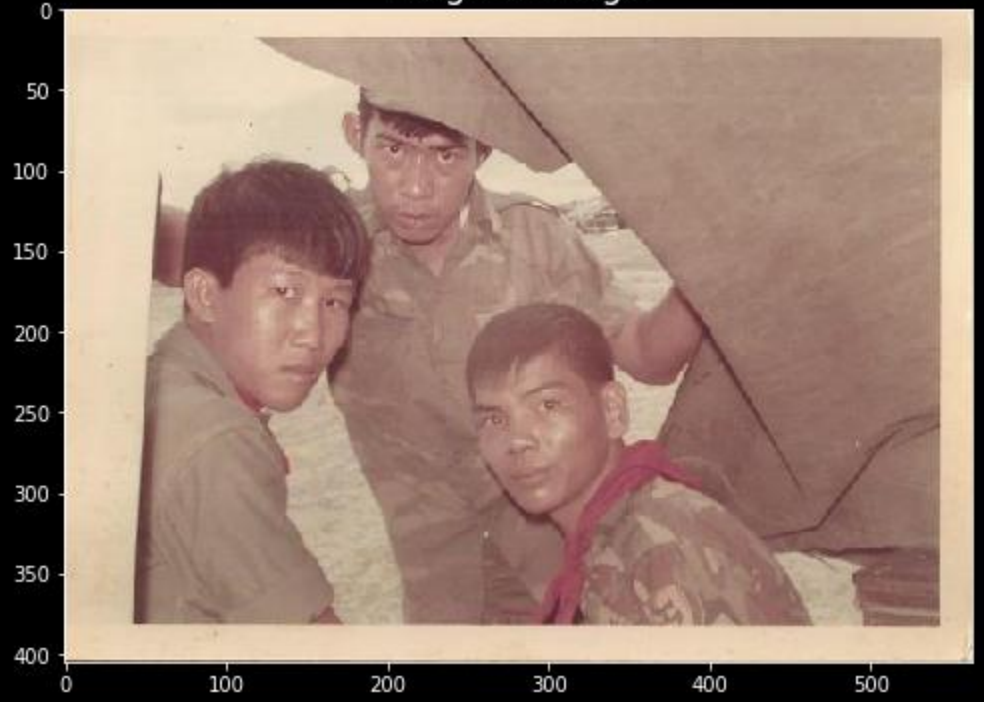
White Patched Image

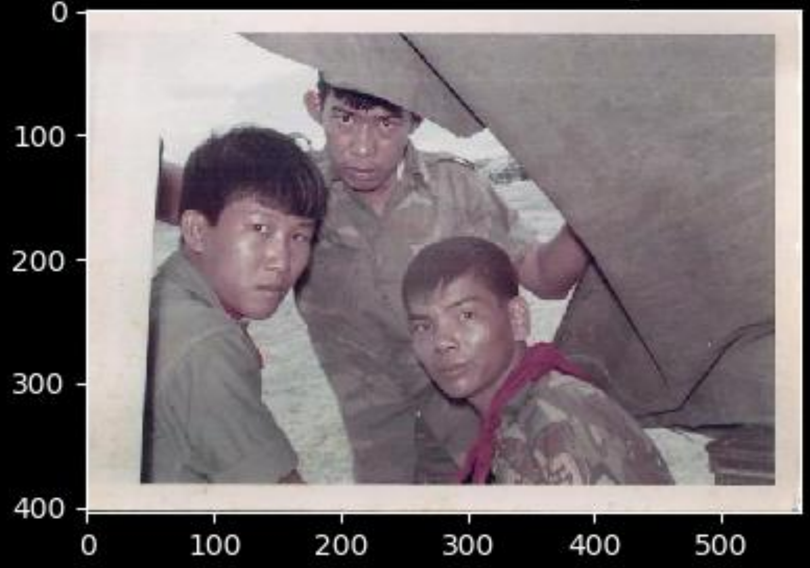


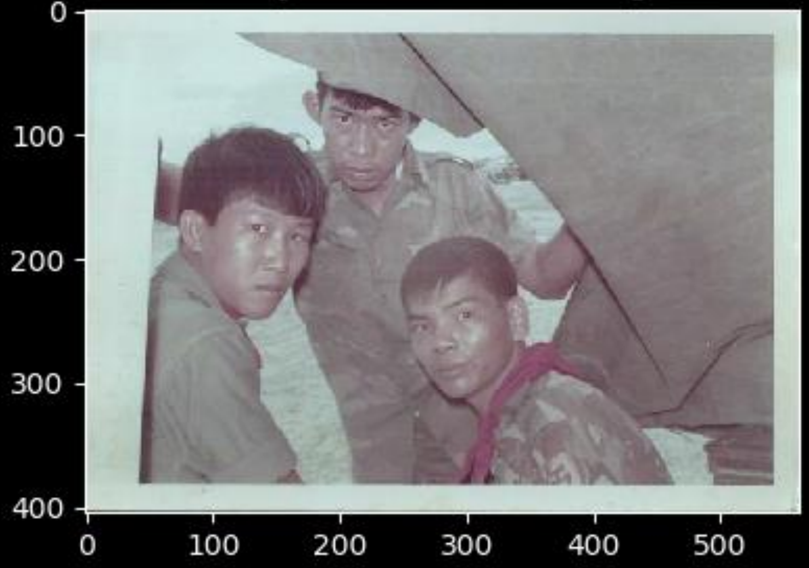
* Comparison

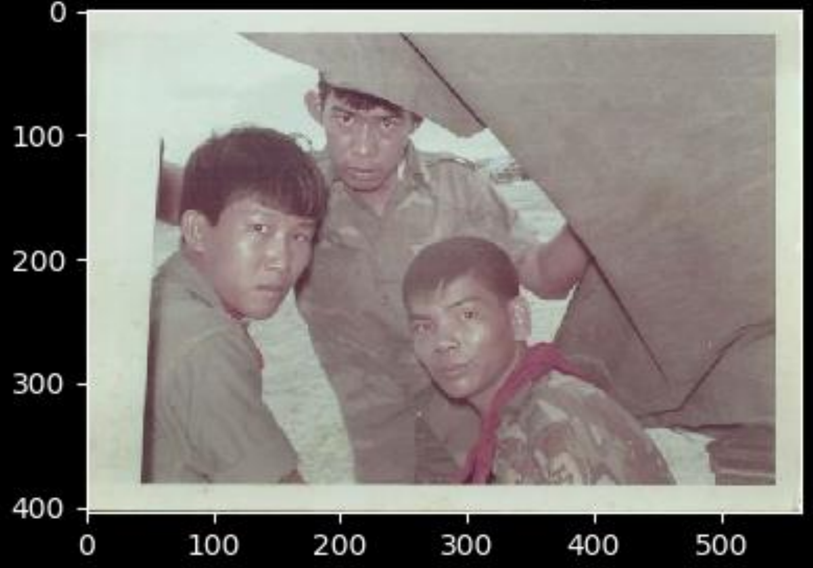
- Most vibrant colors : Contrast Stretched

* Gray World has a bluish tinge similar to a fluorescent lamp balancing constant in digital cameras


Original Image

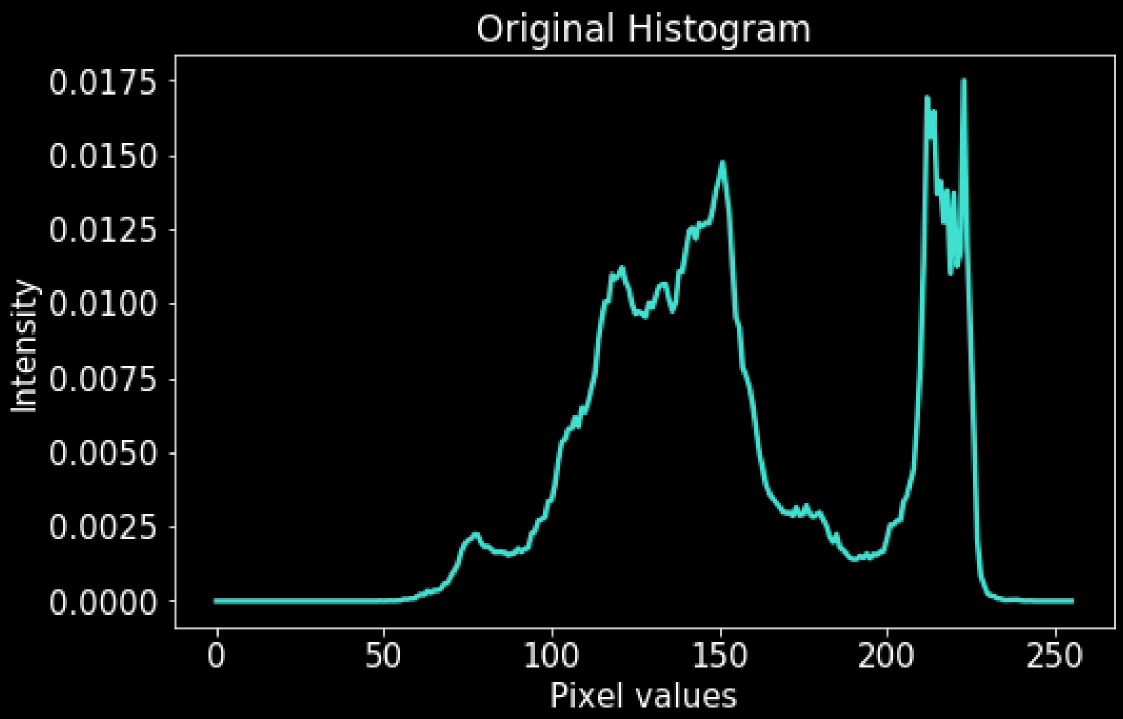

Contrast Stretched Image

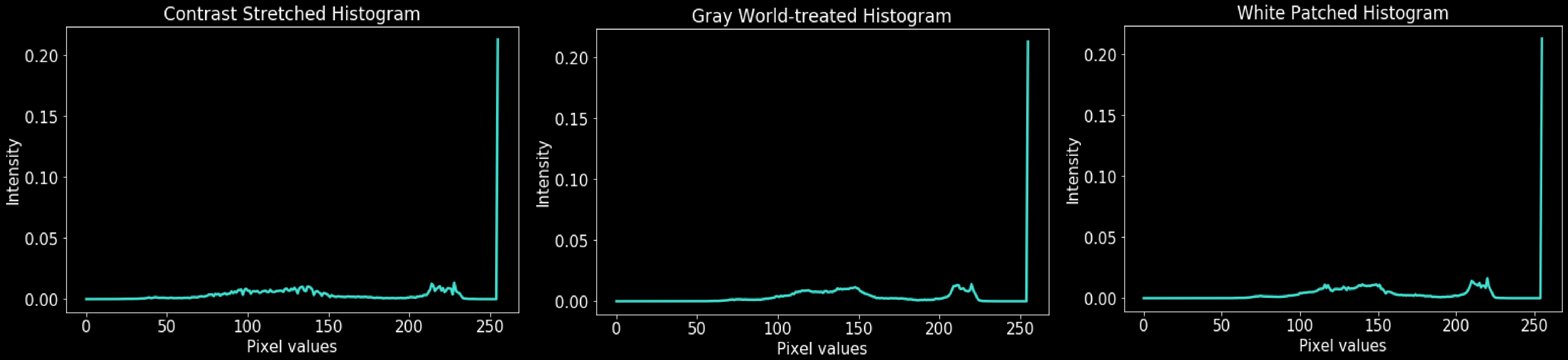

Gray World-treated Image


White Patched Image

* Comparison

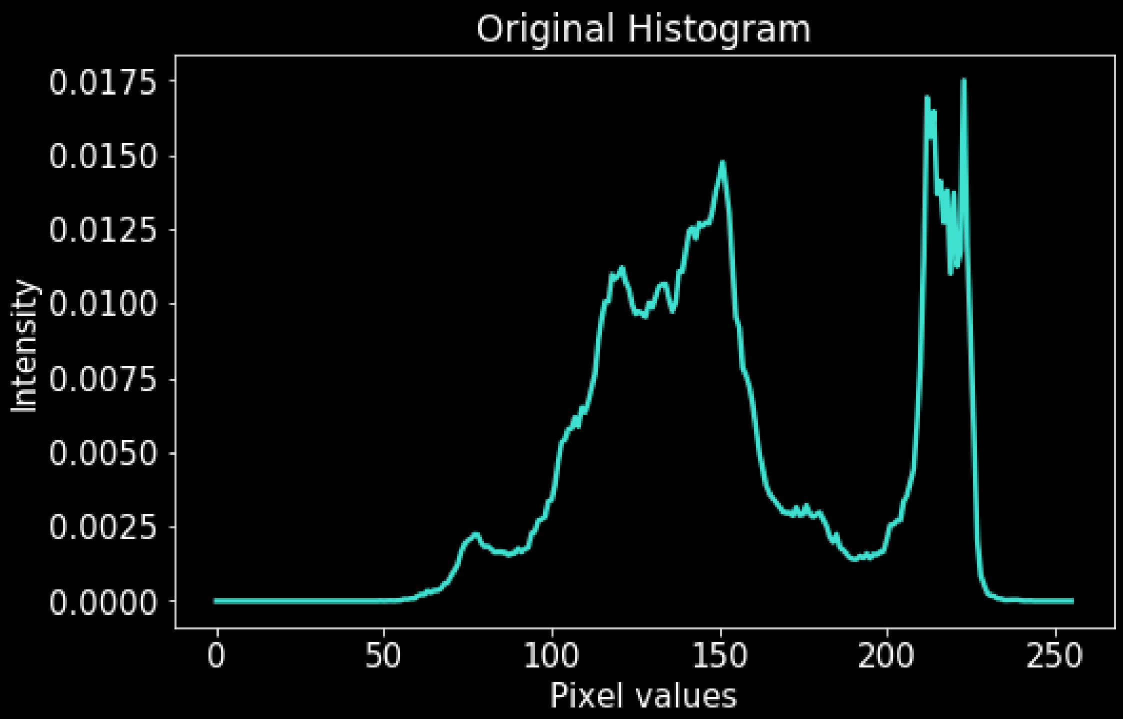
- They all have great shifts to white pixels
- Their difference lie in their corresponding midtone

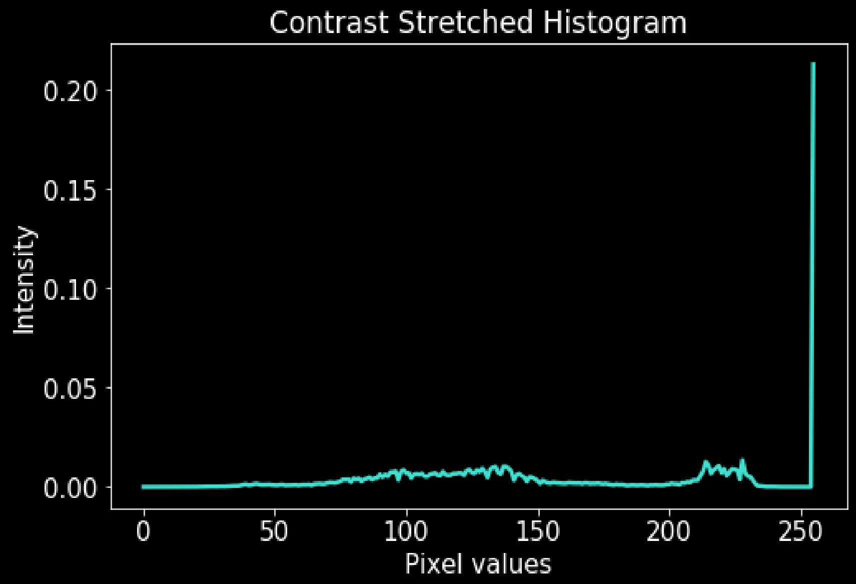


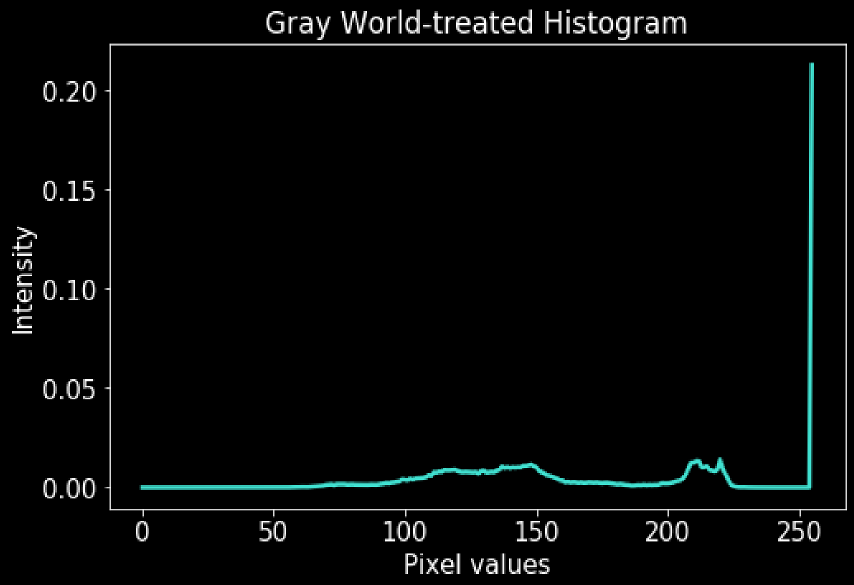


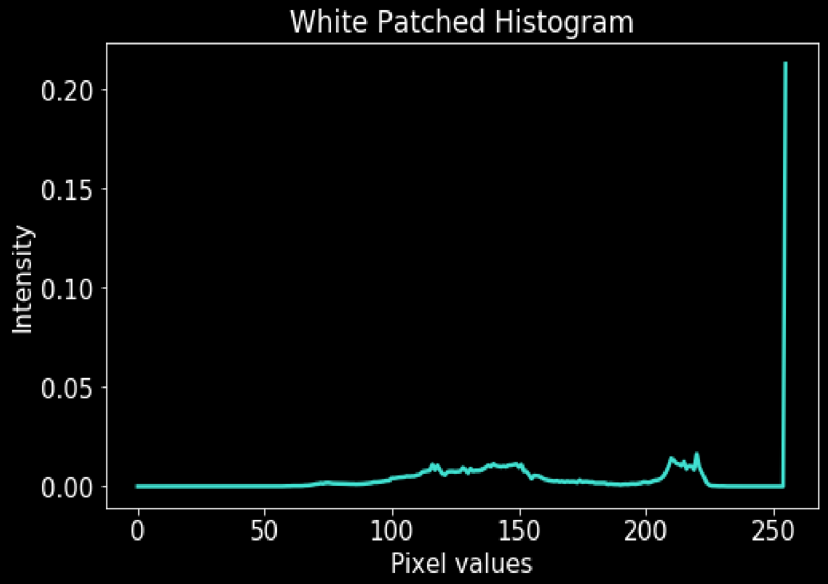
* Comparison

- Gray world has the smoothest curve in the midtones region (approx. 100- 150 px). After all, the average RGB was taken and approximated to have the color gray.



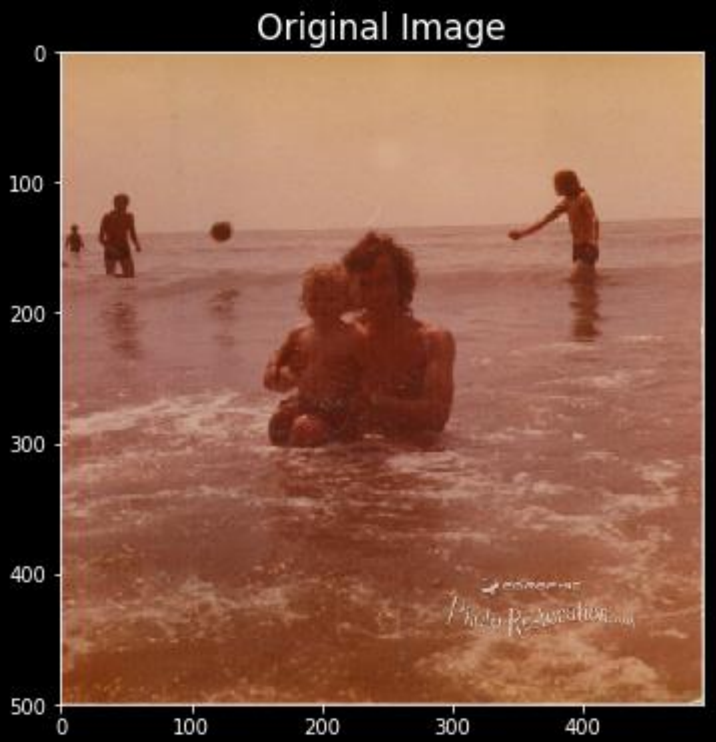


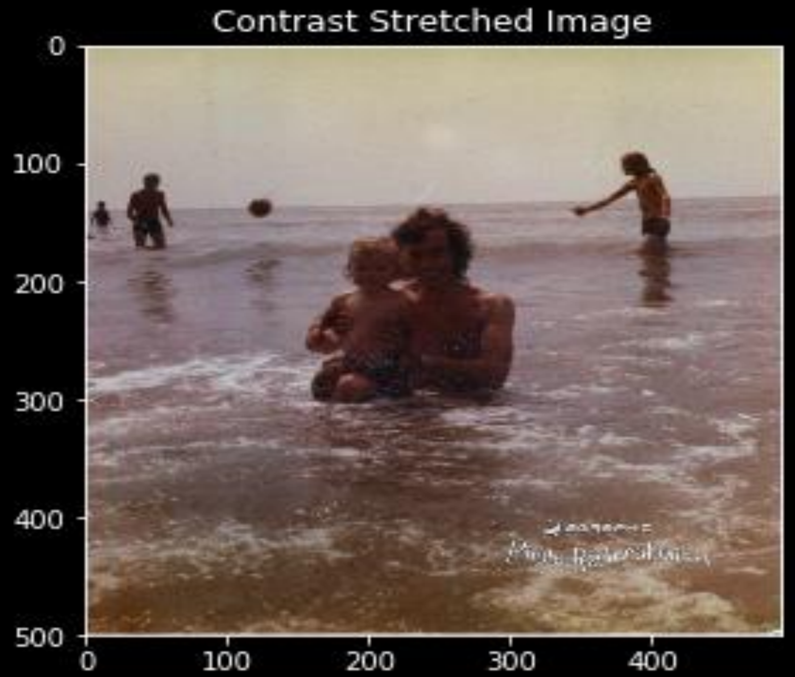


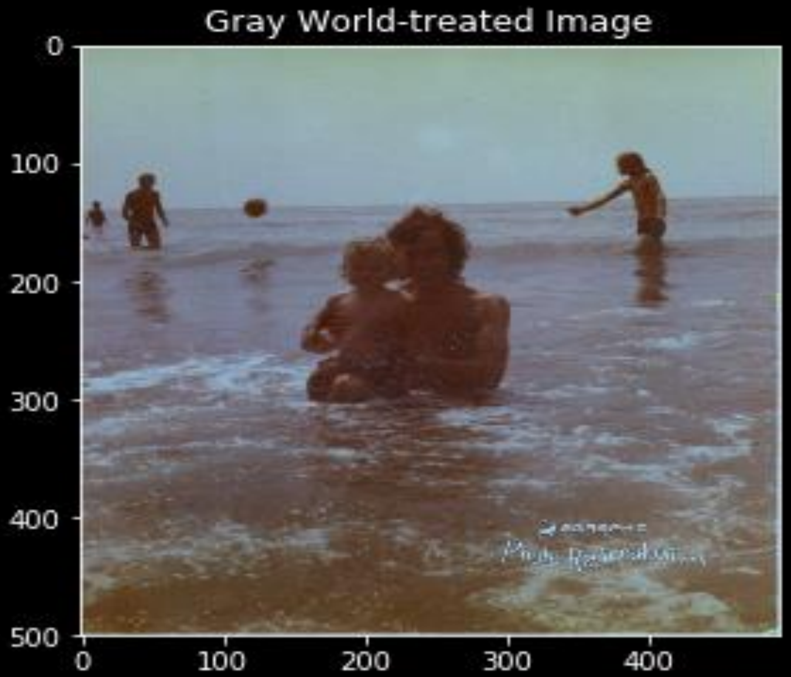


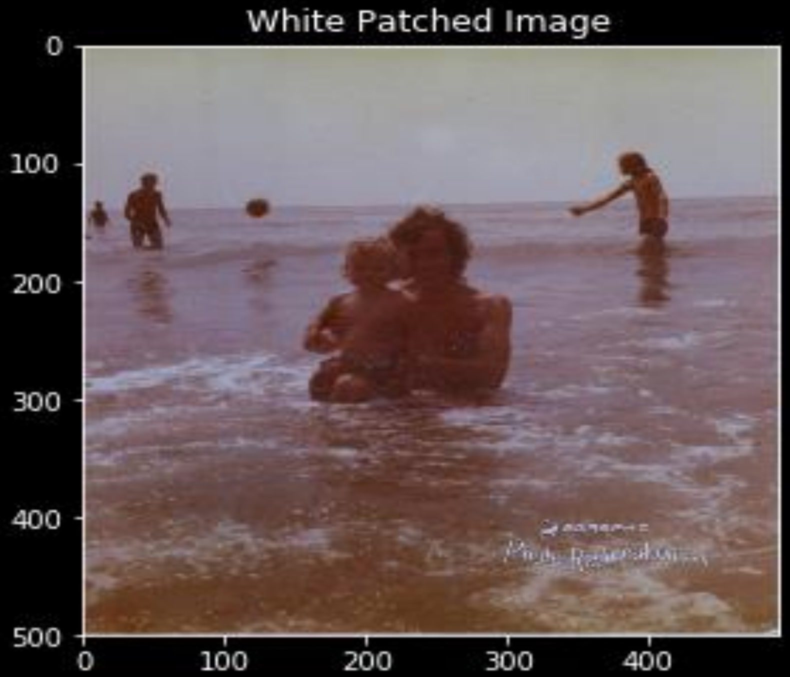
* Other pictures

- Only White Patched Image retained the orange tinge
- Sharpest Image : Contrast Stretch
- Brightest Image : Gray World Algorithm


Original Image


Contrast Stretched Image

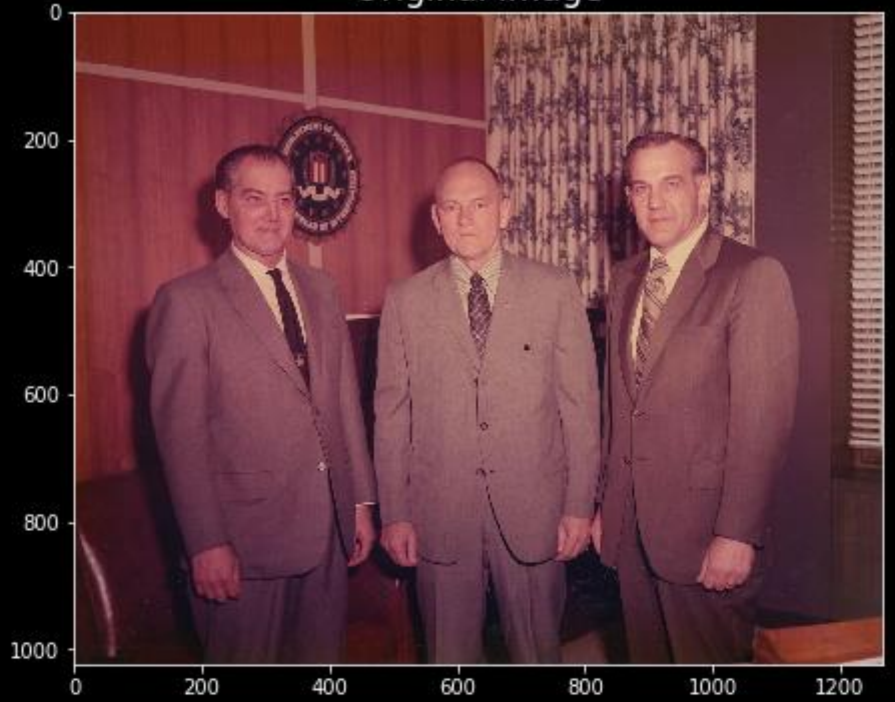

Gray World-treated Image


White Patched Image

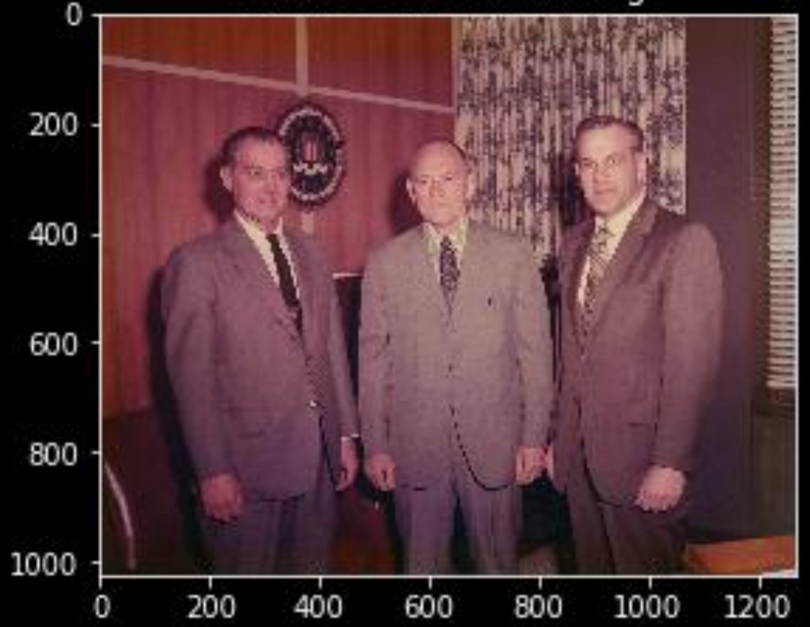
* *Other pictures*

- In the original image, not all regions have faded. The faded region’s boundary at the upper left is obvious.
- Sharpest Image : They all retained similar sharpness

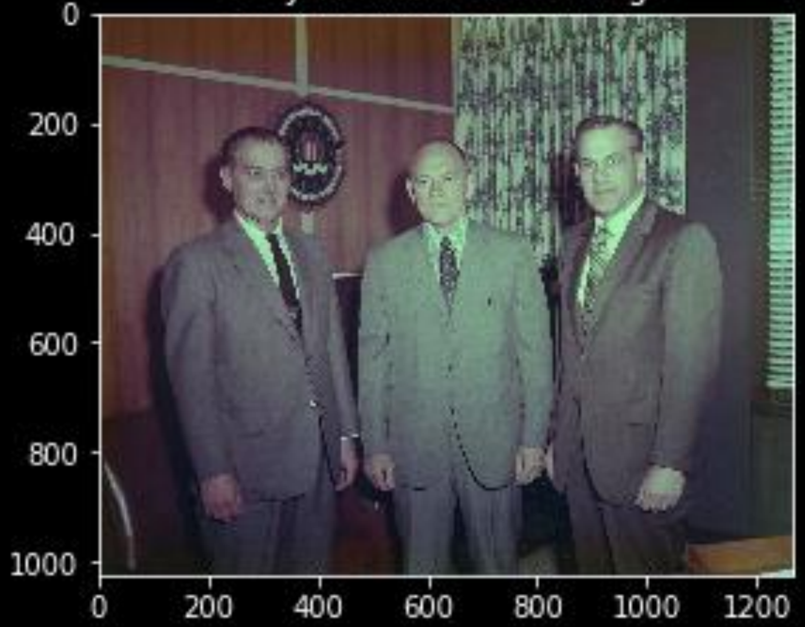
Original Image



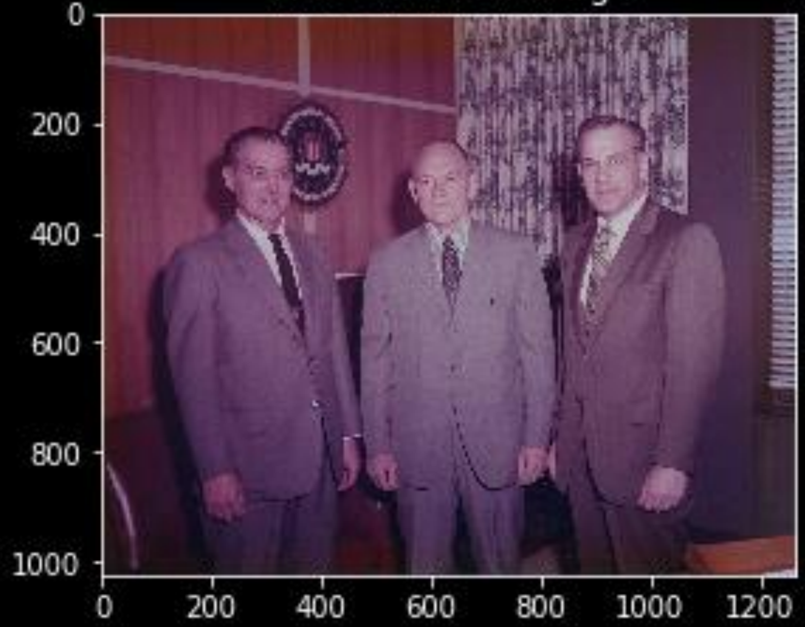
Contrast Stretched Image



Gray World-treated Image

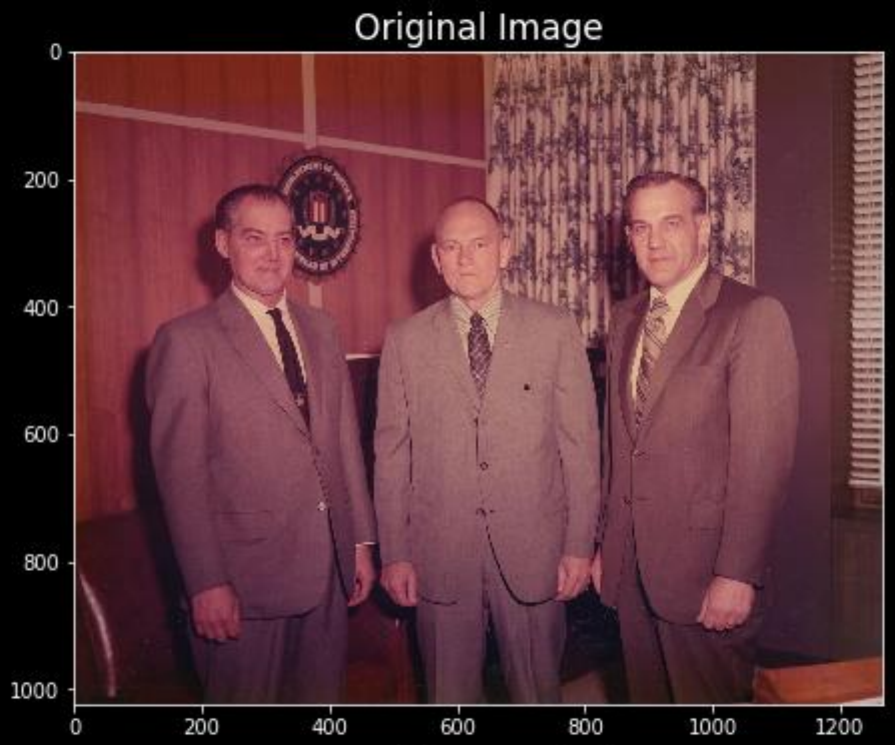


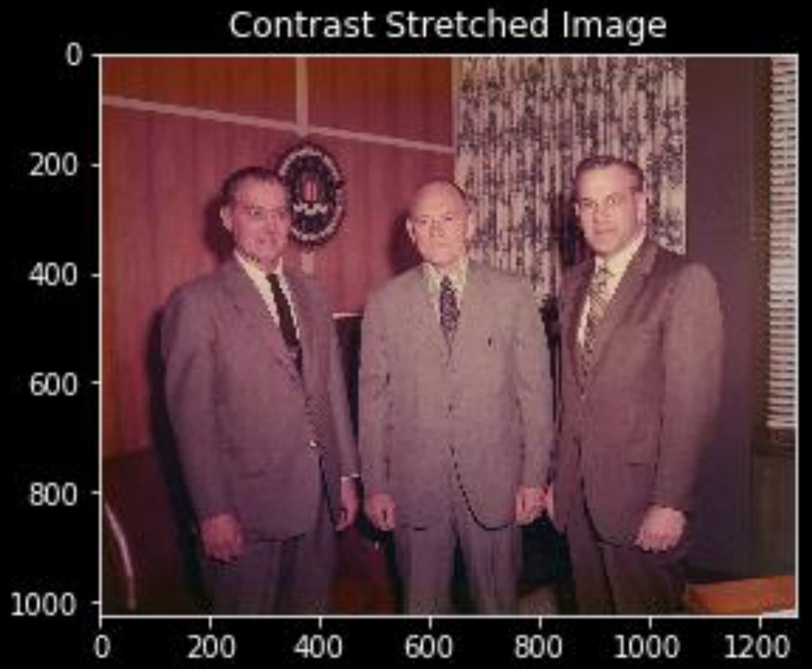
White Patched Image

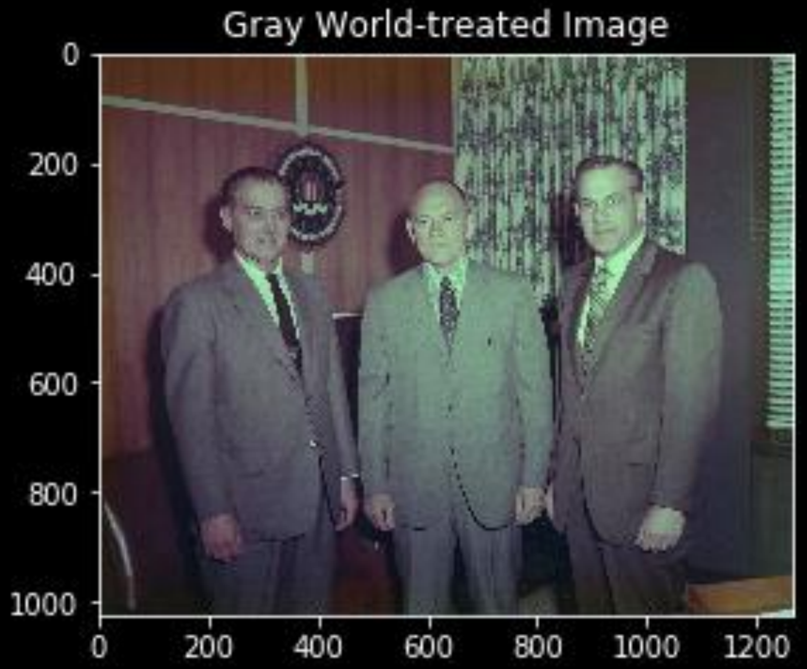


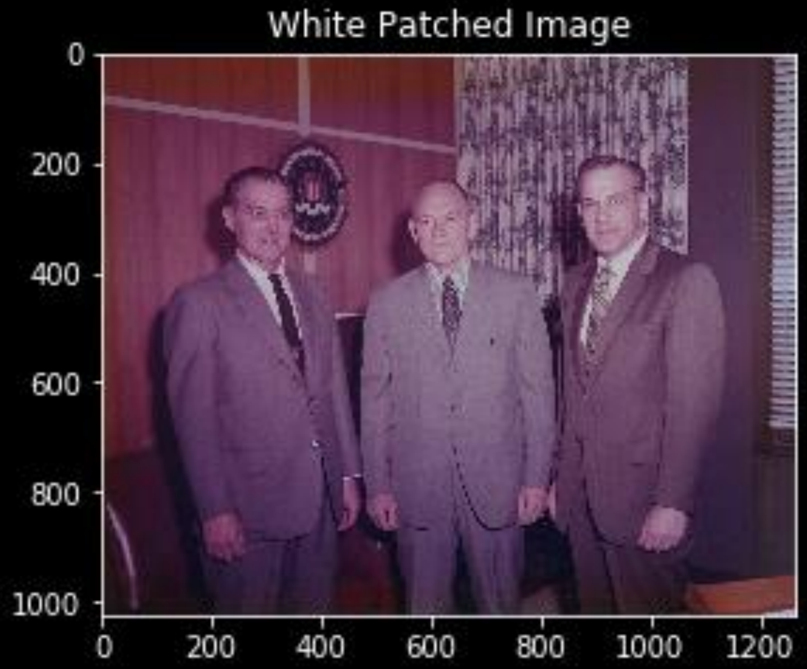
Other pictures

- Brightest Image : Gray World Algorithm (still has that bluish tinge)
- White patched image has brightened but retained the purplish tinge


Original Image


Contrast Stretched Image

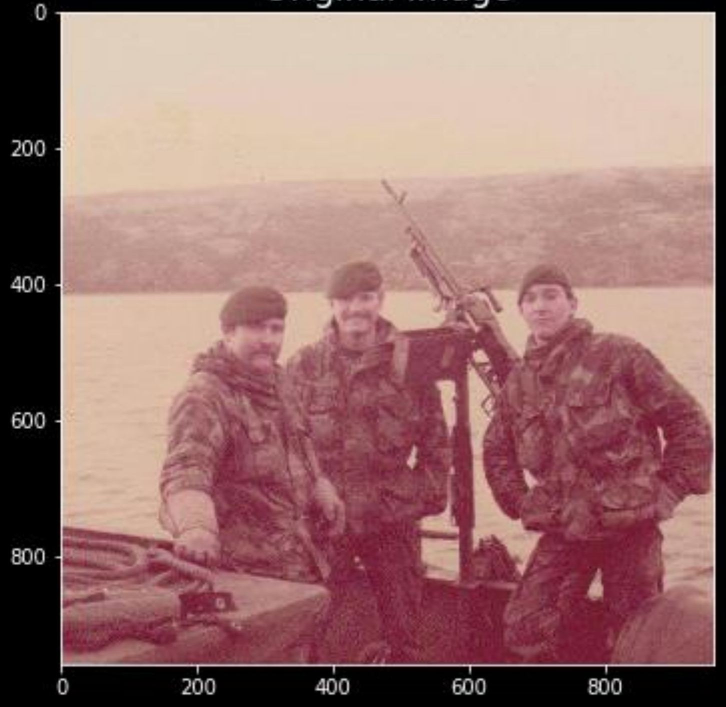

Gray World-treated Image


White Patched Image

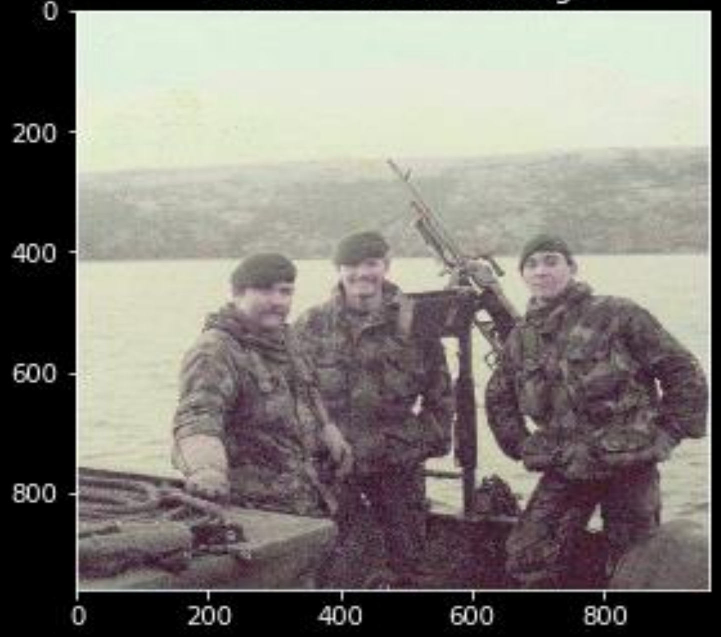
Other pictures

- All methods of restoration has depleted the orange tinge from the original image
- Sharpest Image : Contrast Stretch
- Brightest Image : White Patch Algorithm

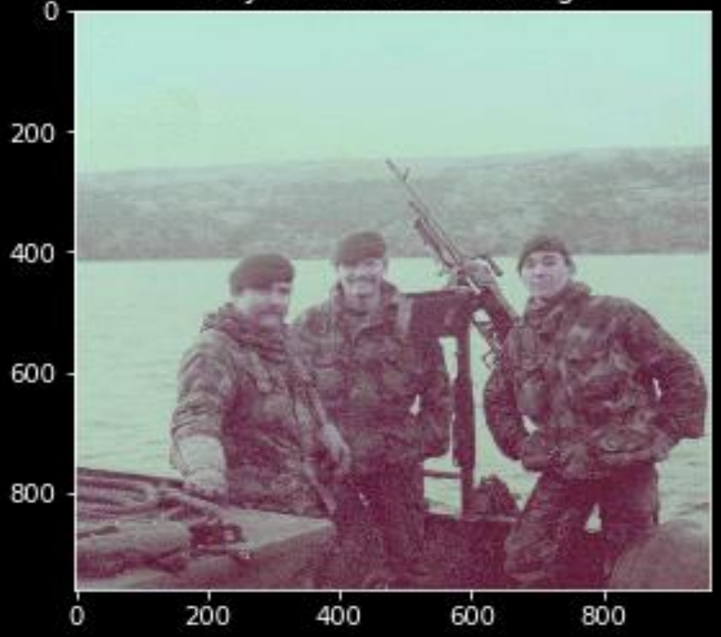
Original Image



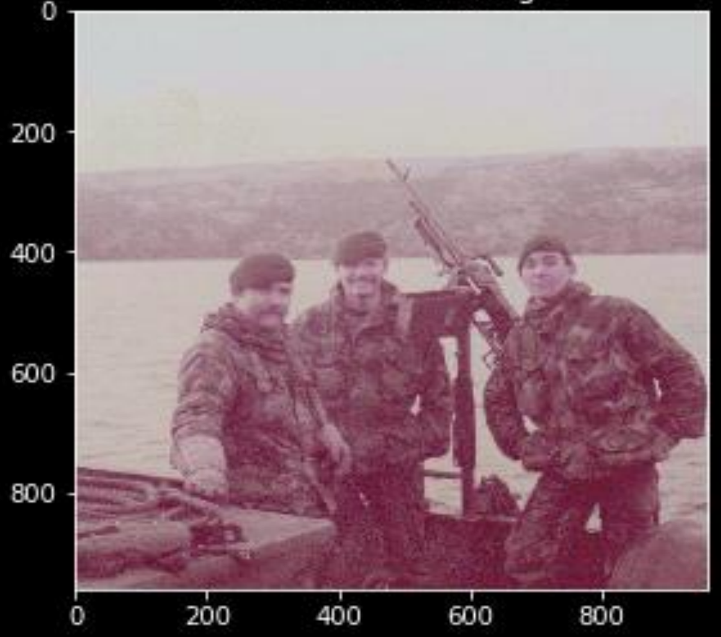
Contrast Stretched Image



Gray World-treated Image



White Patched Image



Other pictures

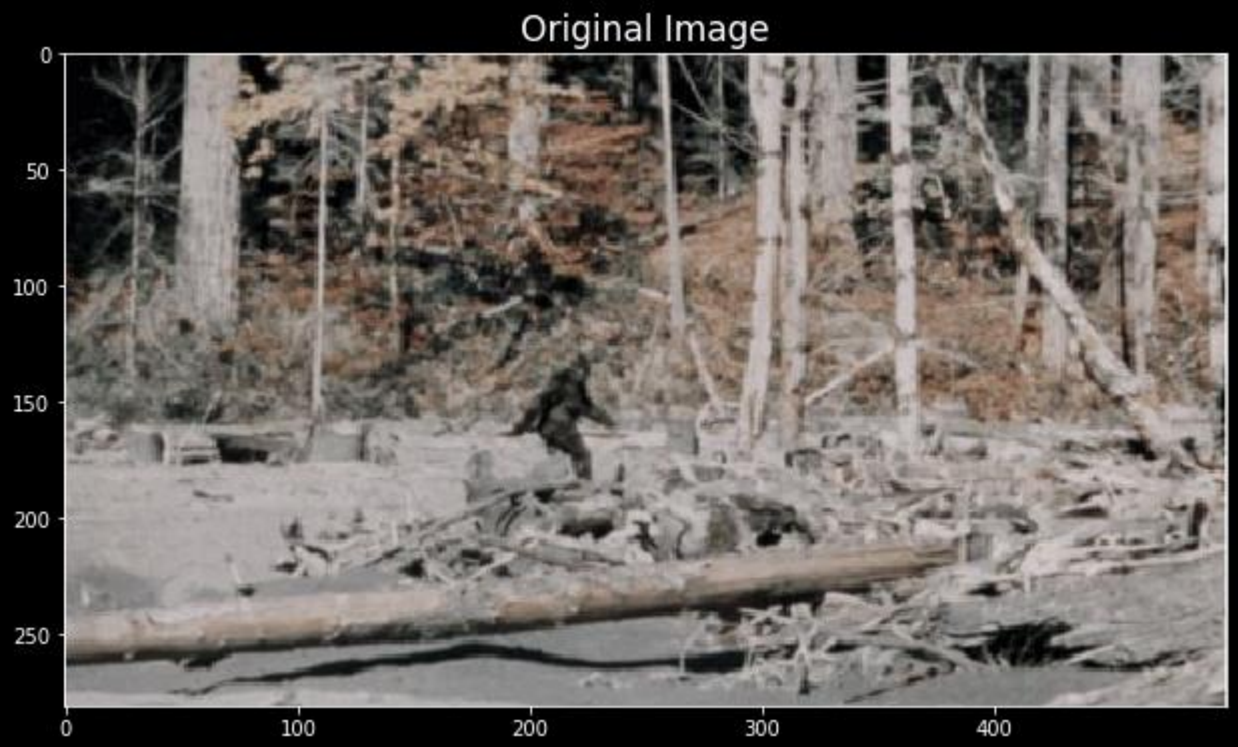
- Concluding from other pictures used, the results differ depending on the original image.
- Preference lies on the user’s taste or goal in restoring the image
- Nevertheless, these different methods in white balancing applied for restoration were very effective in changing old, faded photos.

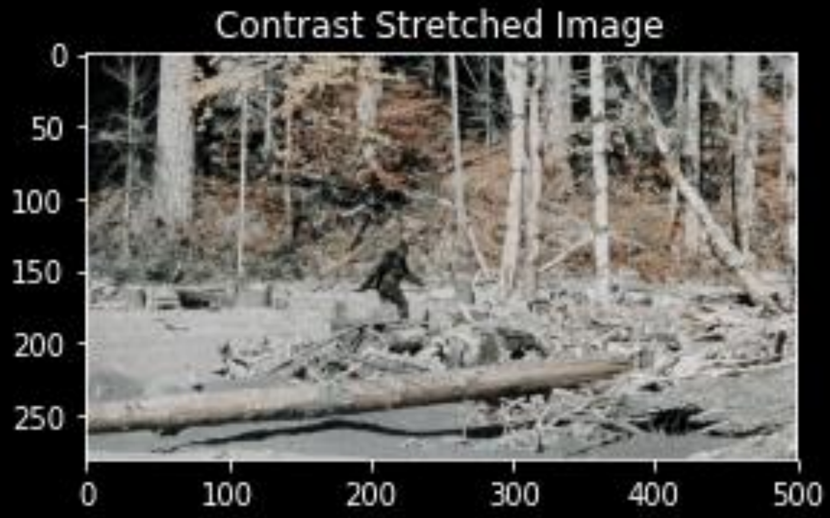
Bonus Step * BIGFOOT

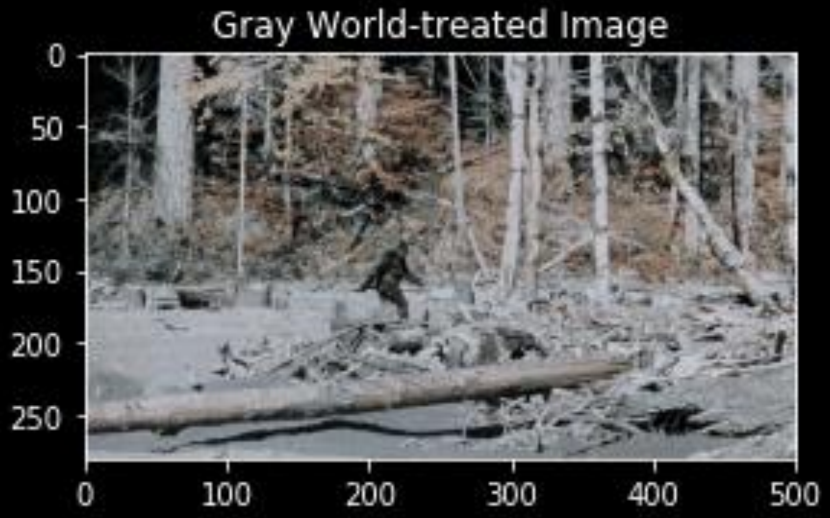
Like what I did in Act. 5, I’m going to use this leaked image of Big Foot and enhance it.

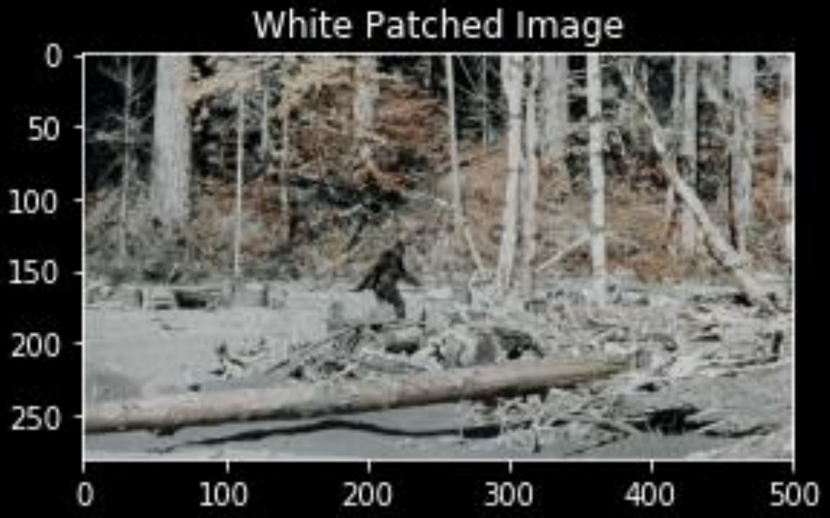
Comparison

- I haven’t gotten much information on the Big Foot itself.
- However, results below confirm the final results for this activity.


Original Image


Contrast Stretched Image


Gray World-treated Image


White Patched Image

References

[1] https://www.photoancestry.com/why-do-my-photos-fade-and-deteriorate.html

Also got my help from:

https://www.codementor.io/innat_2k14/image-data-analysis-using-numpy-opencv-part-1-kfadbafx6

Pictures taken were mostly from Pinterest and Google

* *Pointssss*

- TC : 5
- QP : 5
- IN : 2…..??

- This was so much fun ☺! Thank you!