Image Processing Assignment 10

Color Image Enhancement and Histogram Manipulation

Aditya Chavan
Roll No: 231080019
T.Y B.Tech IT

Date: November 6, 2025

1 Aim

To write a Python program for performing Color Image Enhancement and Histogram Manipulation using OpenCV and Matplotlib.

2 Theory

Image enhancement improves the visual quality of images or modifies them for better analysis. Histogram manipulation techniques such as equalization or CLAHE (Contrast Limited Adaptive Histogram Equalization) are commonly used to enhance contrast.

2.1 Key Concepts

1. **Color Image Enhancement:** Modifying color channels (R, G, B) to improve contrast and visibility.
2. **Histogram Equalization:** Adjusting intensity distribution for better contrast.
3. **CLAHE:** Enhances contrast adaptively across small tiles, avoiding over-enhancement.

2.2 Mathematical Foundation

For histogram equalization:

$$s = T(r) = \int_0^r p_r(w)dw$$

where $p_r(w)$ is the probability density of the image intensity values.

3 Algorithm

3.1 Steps

1. Load the color image.
2. Display and analyze the RGB histograms.
3. Perform histogram equalization on each channel.
4. Display equalized image and new histograms.

5. Apply CLAHE for adaptive enhancement.
6. Compare original, equalized, and CLAHE results.

4 Implementation

The implementation was carried out in Google Colab using Python and OpenCV.

4.1 Required Packages

```
!pip install opencv-python matplotlib numpy
```

4.2 Code Implementation

Google Colab Notebook: Click here to view the Colab Notebook

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
from google.colab.patches import cv2_imshow
```

```
# read and display the color image
img = cv2.imread("2637588.jpg")
# Display
cv2_imshow(img)
```

```
# Plot histograms for R, G, B channels
color = ('r','g','b')
for i,col in enumerate(color):
    hist = cv2.calcHist([img],[i],None,[256],[0,256])
    plt.plot(hist,color = col)
    plt.xlim([0,256])
plt.title('Histogram of Original Image')
plt.show()
```

```
# Split channels
r, g, b = cv2.split(img)
```

```
# Equalize each channel separately
r_eq = cv2.equalizeHist(r)
g_eq = cv2.equalizeHist(g)
b_eq = cv2.equalizeHist(b)
```

```
# Merge equalized channels
equalized_img = cv2.merge((r_eq, g_eq, b_eq))
```

```
# Display result
plt.subplot(1,2,1)
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(img_rgb)
plt.title('Original Image')
plt.axis('off')

plt.subplot(1,2,2)
equalized_img_rgb = cv2.cvtColor(equalized_img, cv2.COLOR_BGR2RGB
    )
plt.imshow(equalized_img_rgb)
plt.title('Histogram Equalized Image')
plt.axis('off')
plt.show()
```

```
# Plot histograms for Equalised R, G, B channels
color = ('r','g','b')
for i,col in enumerate(color):
    hist = cv2.calcHist([equalized_img_rgb],[i],None
        ,[256],[0,256])
    plt.plot(hist,color = col)
    plt.xlim([0,256])
plt.title('Histogram Equalised')
plt.show()
```

```
# Create CLAHE object
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))

# Apply CLAHE on each channel
r_clahe = clahe.apply(r)
g_clahe = clahe.apply(g)
b_clahe = clahe.apply(b)

# Merge CLAHE enhanced channels
clahe_img = cv2.merge((r_clahe, g_clahe, b_clahe))
```

```
# Display result
plt.subplot(1,2,1)
plt.imshow(img_rgb)
plt.title('Original Image')
plt.axis('off')

plt.subplot(1,2,2)
clahe_img_rgb = cv2.cvtColor(clahe_img, cv2.COLOR_BGR2RGB)
plt.imshow(clahe_img_rgb)
plt.title('CLAHE Enhanced Image')
plt.axis('off')
plt.show()
```

5 Results and Analysis

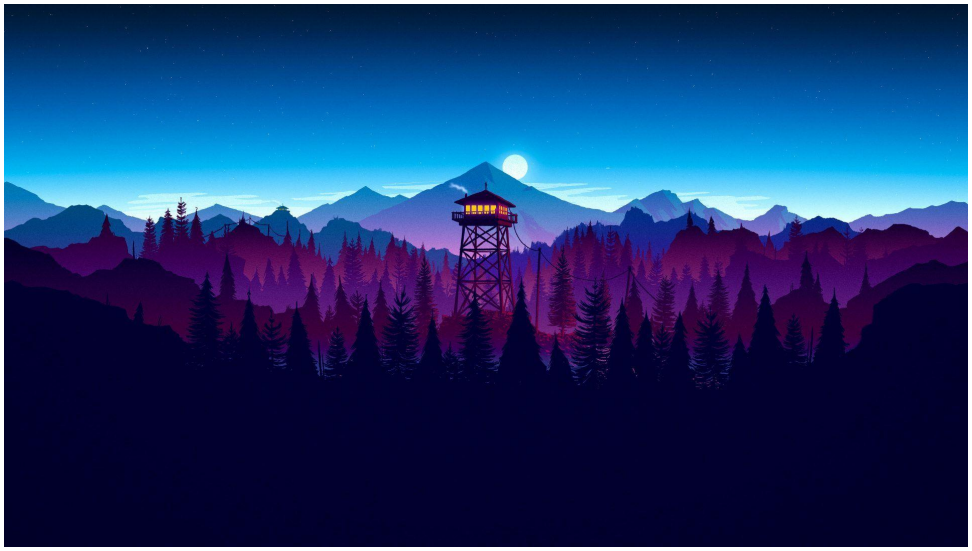

Figure 1: Original Input Image

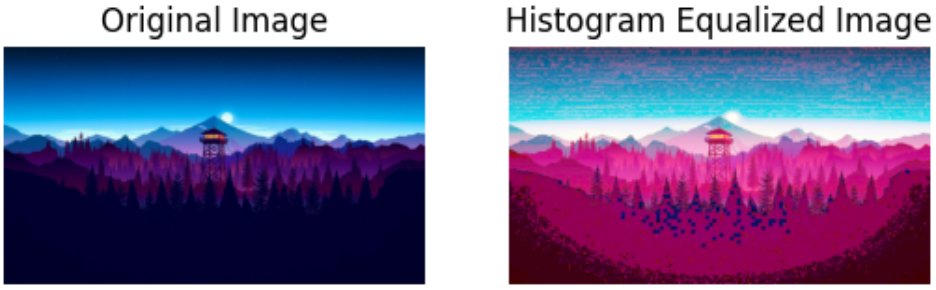

Figure 2: Histogram Equalized Image

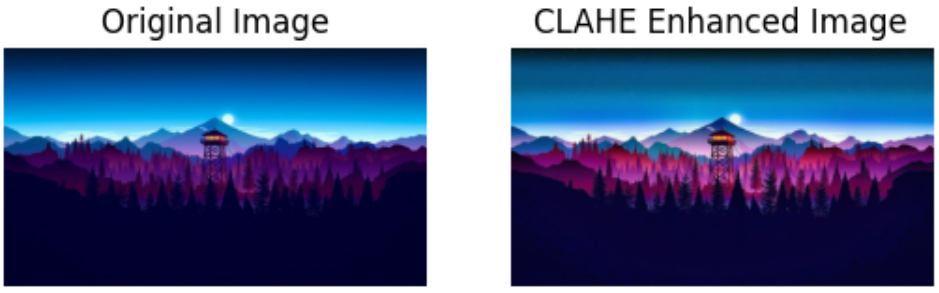

Figure 3: CLAHE Enhanced Image

6 Conclusion

This experiment demonstrated color image enhancement using Histogram Equalization and CLAHE, improving global and local contrast. These techniques are vital for enhancing visuals in medical imaging, photography, and computer vision.