231501037

EXP NO: 03
DATE: 18-07-2025

Histogram processing and Equalization.

Aim: To Implement Histogram processing and Equalization.

Algorithm:

1. Convert image to grayscale.
2. Compute histogram using cv2.calcHist().
3. Normalize and plot the histogram.
4. Apply histogram equalization using cv2.equalizeHist().
5. Compare before and after histograms.
6. Display results.

Code:

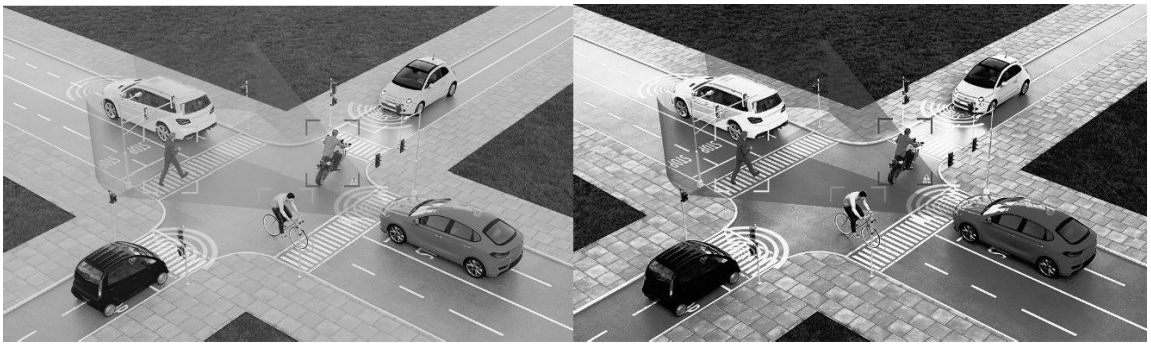
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from google.colab.patches import cv2_imshow

# Read image
img = cv2.imread('/content/drive/MyDrive/input.jpg')

# Check if image is loaded
if img is None:
    print("Error: Could not load image. Please check the file path.")
else:
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Histogram
    hist = cv2.calcHist([gray], [0], None, [256], [0,256])
```

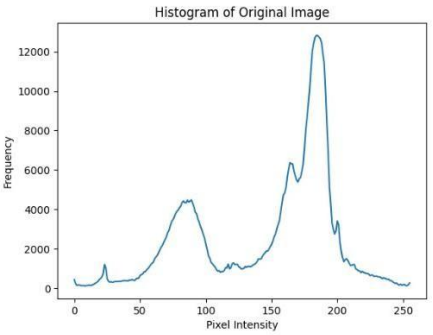
231501037

```
# Equalization
equalized = cv2.equalizeHist(gray)

# CLAHE
clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
clahe_img = clahe.apply(gray)

# Show images
cv2_imshow(gray)
cv2_imshow(equalized)
cv2_imshow(clahe_img)

# Plot histogram
plt.figure()
plt.title("Histogram of Original Image")
plt.xlabel("Pixel Intensity")
plt.ylabel("Frequency")
plt.plot(hist)
plt.show()

# cv2.waitKey(0)
# cv2.destroyAllWindows()

# Save results
cv2.imwrite("equalized.jpg", equalized)
cv2.imwrite("clahe.jpg", clahe_img)
```

231501037

Output:





Result: Thus, Histogram processing and Equalization implemented successfully.