Lab Experiment 7 Date: 6/03/2025

Khushi Malhotra || AP22110010450

Ques: Write a program to improve contrast of an image using histogram equalization. The prototype of the function is as below: histogram_equalisation(input_Image, no_of_bins); The function should return the enhanced image. Consider two low contrast input images. Study the nature of the output image quality in each case by varying the number of bins. give aim of this question and then python code as well.

Aim - The objective of this task is to improve the contrast of a low-contrast image using Histogram Equalization. This technique redistributes the intensity values of an image so that they span a wider range, making the image visually clearer. By varying the number of bins, we can study how the quality of the output image changes.

Theory - 1. Histogram Equalization (HE) is a contrast enhancement technique that redistributes pixel intensity values to improve visibility.

2. It spreads out the most frequent intensity values across the full intensity range (0-255 for grayscale images).
3. The process involves computing the histogram, calculating the cumulative distribution function (CDF), normalizing the CDF, and mapping pixel values accordingly.
4. HE enhances contrast by making dark areas darker and bright areas brighter.
5. It is widely used in medical imaging, remote sensing, automotive vision, and surveillance systems.
6. The number of bins affects the quality of enhancement, with fewer bins causing blocky artifacts and more bins providing smoother transitions.
7. HE works best for images with poor contrast but can introduce noise amplification in some cases.
8. It is not directly applicable to color images unless applied separately to each color channel.
9. Adaptive Histogram Equalization (AHE) and Contrast Limited Adaptive Histogram Equalization (CLAHE) are advanced versions that work on localized regions to avoid over-enhancement.
10. HE is a simple yet effective technique for improving image quality in low-contrast scenarios.

```
import cv2
import numpy as np
```

```
import matplotlib.pyplot as plt
import os

def histogram_equalisation(input_Image, no_of_bins):
    """
    Perform histogram equalization on the input image.

    Parameters:
        input_Image (numpy.ndarray): Grayscale input image.
        no_of_bins (int): Number of bins for histogram equalization.

    Returns:
        numpy.ndarray: Enhanced image.
    """
    # Check if the image is loaded properly
    if input_Image is None:
        print("Error: Image not loaded properly!")
        return None

    # Convert to grayscale if not already
    if len(input_Image.shape) == 3:
        input_Image = cv2.cvtColor(input_Image, cv2.COLOR_BGR2GRAY)

    # Compute histogram
    hist, bins = np.histogram(input_Image.flatten(), no_of_bins, [0, 256])

    # Compute cumulative distribution function (CDF)
    cdf = hist.cumsum()
    cdf_normalized = cdf * 255 / cdf[-1]  # Normalize CDF

    # Perform histogram equalization
    equalized_image = np.interp(input_Image.flatten(), bins[:-1], cdf_normalized)
    equalized_image = equalized_image.reshape(input_Image.shape).astype(np.uint8)

    return equalized_image

# Function to load an image safely
def load_image(image_path):
    if not os.path.exists(image_path):
        print(f"Error: File '{image_path}' not found!")
        return None
    image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    if image is None:
        print(f"Error: Could not load image '{image_path}'!")
    return image

# Load two low contrast images safely
image1_path = '/dhoni.jpg'
image2_path = '/dhoni.jpg'

image1 = load_image(image1_path)
image2 = load_image(image2_path)

# Check if images were loaded before proceeding
if image1 is None or image2 is None:
```