Title: Histogram Equalization
Subtitle: A Technique for Image Enhancement

Histogram Equalization

- Histogram Equalization is a technique used in image processing to enhance the contrast of an image.
- The idea is to redistribute the intensity levels (pixel values) of the image so that the range of intensities (brightness levels) is more evenly spread out, especially in images withpoor contrast.
- This makes it easier to see details in dark or bright areas of the image.

How Histogram Equalization Works

- 1. Histogram: An image’s histogram is a graphical representation of the distribution of pixel intensities (brightness levels) in the image. It shows how many pixels in the image have a particular intensity value.
- 2. Cumulative Distribution Function (CDF): The CDF is calculated by accumulatin the frequency of each intensity level in the histogram.
- 3. Mapping: The pixel values are mapped to new intensity values based on the CDF, which spreads out the most frequent pixel values over the full range of availableintensities.

Step-by-Step Process:

1. **Find the Histogram**: First, compute the histogram of the image. This shows the frequency of each pixel intensity (from 0 to 255 for an 8-bit grayscale image).
2. **Compute the CDF:** Calculate the cumulative distribution function of the histogram. This function represents the cumulative sum of pixel frequencies up to each intensity level.
3. **Normalize the CDF:** The CDF is normalized by dividing each value by the total number of pixels in the image. This scales the CDF values between 0 and 1.
4. **Map the Pixel Values:** For each pixel intensity, map it to a new value based on the normalized CDF. This spreads the pixel values over the entire intensity range (0–255), resulting in better contrast.
5. **Recreate the Image:** Finally, the image is transformed by replacing each pixel value with its corresponding new value, creating an enhanced image.

Example of Histogram Equalization:

Consider a grayscale image where most pixel values are clustered around a lower intensity range, making it look dark and lacking contrast.

Original Histogram:

makefile

Copy

Intensity:	0	50	100	150	200	250
Frequency:	10	30	100	50	10	5

This histogram shows that most pixels have values around 100–150, meaning the image is mostly dark with little contrast.

After Histogram Equalization:

The histogram will be adjusted such that the intensity values are spread more evenly across the entire 0-255 range.

After equalization

makefile

Copy

Intensity:	0	50	100	150	200	250
Frequency:	5	10	30	100	50	10

The pixel values will now span a wider range, improving the contrast and making both dark and bright areas more distinguishable.

Resulting Image:

Before: A dark, poorly contrasted image.

After: A more evenly distributed image with enhanced contrast, where details in both the dark and light areas more clearly.

When to Use Histogram Equalization:

 Low-Contrast Images: Images where the features are hard to distinguish due to poor contrast (e.g., an image taken in dim lighting).

 Medical Imaging: Enhancing medical scans (like X-rays) to make fine details more visible.

 Satellite Imaging: Improving the visibility of terrain or structures in satellite images.

Benefits of Histogram Equalization:

 Improved Contrast: Increases the contrast, especially in images with poor visibility of details.

 Better Feature Detection: Makes it easier for algorithms (like object detection or edge detection) to find features in the image.

 Visual Clarity: Makes the image more visually clear to the human eye.

Limitations:

 Over-enhancement: If applied to an already well-contrasted image, it might introduce noise and make the image look unnatural.

 Color Images: Histogram equalization works best on grayscale images. For color images, it can be applied to individual channels (R, G, B), but it may result in unnatural color shifts.

You are signed in | Histogram Equaliz | Home | Gray1 Scale | Jupyter Server | Jupyter Server | Brightness

localhost:8888/notebooks/Desktop/Image%20practical/Gray1%20Scale.ipynb?

jupyter Gray1 Scale Last Checkpoint: 5 minutes ago

File Edit View Run Kernel Settings Help Not Trusted

Code JupyterLab Python [conda env:base] *

[ ]:

[3]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt


# Step 1: Load the image in grayscale
img1 = cv2.imread('D://SJC//Image Processing//tomato.jpg')  # Ensure the image is in grayscale
img= cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)

# Step 2: Calculate the histogram of the original image
original_histogram = cv2.calcHist([img], [0], None, [256], [0, 256])

# Step 3: Apply Histogram Equalization
equalized_img = cv2.equalizeHist(img)

# Step 4: Calculate the histogram of the equalized image
```

Type here to search 20°C Haze ENG 10:04 31-01-2025

jupyter Gray1 Scale Last Checkpoint: 6 minutes ago


```
# Step 4: Calculate the histogram of the equalized image
equalized_histogram = cv2.calcHist([equalized_img], [0], None, [256], [0, 256])

# Step 5: Plot the original and equalized images along with their histograms
plt.figure(figsize=(10, 6))

# Original Image and its Histogram
plt.subplot(2, 2, 1)
plt.imshow(img, cmap='gray')
plt.title("Original Image")
plt.axis('off')

plt.subplot(2, 2, 2)
plt.plot(original_histogram)
plt.title("Original Histogram")

# Equalized Image and its Histogram
```

jupyter Gray1 Scale Last Checkpoint: 7 minutes ago


```
plt.subplot(2, 2, 2)
plt.plot(original_histogram)
plt.title("Original Histogram")

# Equalized Image and its Histogram
plt.subplot(2, 2, 3)
plt.imshow(equalized_img, cmap='gray')
plt.title("Equalized Image")
plt.axis('off')

plt.subplot(2, 2, 4)
plt.plot(equalized_histogram)
plt.title("Equalized Histogram")

# Show the plots
plt.tight_layout()
plt.show()
```

jupyter Gray1 Scale Last Checkpoint: 7 minutes ago

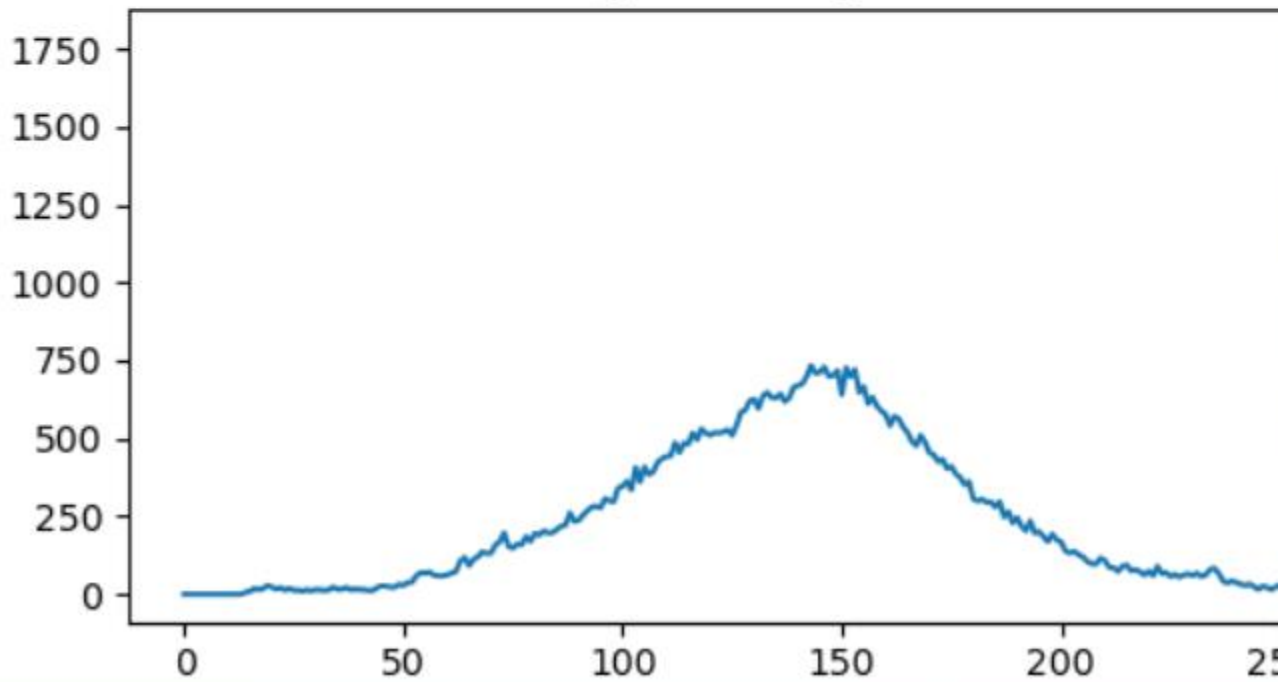
File Edit View Run Kernel Settings Help Not

Code JupyterLab Python [conda env:base] *

```
# Show the plots
plt.tight_layout()
plt.show()
```

Original Image

Original Histogram

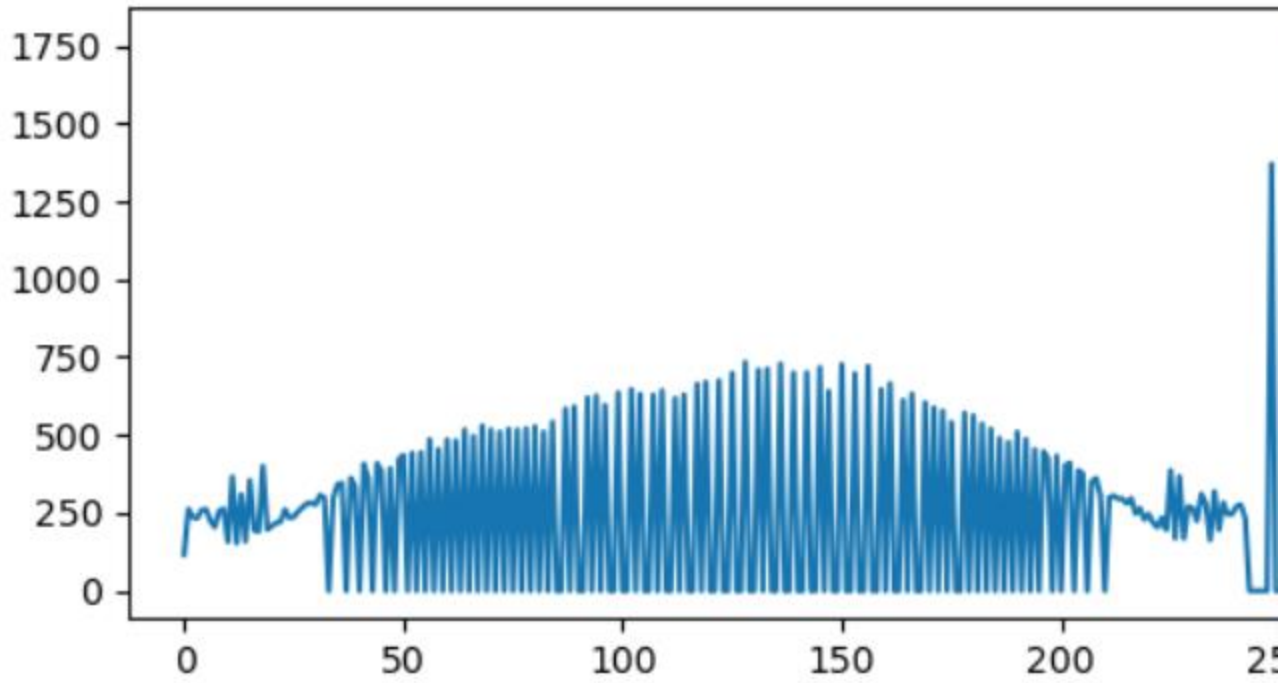


Type here to search 20°C Haze ENG 10:
31-01-

jupyter **Gray1 Scale** Last Checkpoint: 7 minutes ago

Equalized Image

Equalized Histogram



[ ]: