**Assignment No. 2**

**Problem Statement**: Read an image, plot its histogram then do histogram equalization. Comment about the result. Implement various spatial domain and frequency domain filters.

**Aim:** To read a grayscale image, plot its histogram, perform histogram equalization for contrast enhancement, and implement various spatial domain and frequency domain filters for image enhancement and analysis.

## **Objective:**

1. To study the concept of image histogram and its importance in image processing.
2. To apply histogram equalization for contrast improvement.
3. To implement spatial domain filters (smoothing and sharpening) to enhance image details.
4. To implement frequency domain filters (low-pass and high-pass) using Fourier Transform.
5. To analyze and compare the results of spatial and frequency domain filtering.

## **Outcomes:**

1. Understand histogram equalization and its effect on image contrast.
2. Gain practical experience with spatial domain filtering techniques.
3. Implement frequency domain filters using Fourier Transform.
4. Analyze and compare the impact of different image enhancement techniques.

## **Theory**

### **Histogram & Histogram Equalization**

* + Image Histogram: A graph representing the distribution of pixel intensities in an image.
  + Histogram Equalization: A technique to spread out pixel intensities more evenly, improving image contrast.

**Formula**:

# sk = 𝑟𝑜𝑢𝑛𝑑 ( (𝐿 − 1)

𝑘

∑

𝑗=0

# pr (rj) )

where,

sk = output intensity level

L = total number of intensity levels

Pr (rj) = probability of intensity level rj

### **Spatial Domain Filtering**

* + Operates directly on pixels.
  + A filter (mask/kernel) is convolved with the image.

### **Formula:**

𝑎

# g(x,y) = ∑

𝑏

∑ 𝑤(𝑠, 𝑡) . 𝑓(𝑥 − 𝑠, 𝑦 − 𝑡)

𝑠=−𝑎 𝑡=−𝑏

where,

f(x,y) = input image g(x,y) = output image

w(s,t) = filter mask / kernel

### **Examples:**

* + Gaussian Blur (Smoothing) → reduces noise.
  + Sharpening (Laplacian/High-boost) → enhances edges.

### **Frequency Domain Filtering**

* + Based on Fourier Transform, where the image is represented in terms of frequency components.
  + High frequencies → edges, noise, ﬁne details.
  + Low frequencies → smooth background, general shape.

**Discrete Fourier Transform Formula:**

# X(k) =

𝑁−1

−𝑗2Π𝑘𝑛/𝑁

∑ 𝑥(𝑛) 𝑒

𝑛=0

**Inverse Discrete Fourier Transform:**

# x(n) = 1

𝑁−1

𝑗2Π𝑛𝑘/𝑁

𝑁 ∑ 𝑋(𝑘) 𝑒

𝑘=0

* + **Low-pass filter** → retains low frequencies, removes high frequencies → smoothing.
  + **High-pass filter** → retains high frequencies, removes low frequencies → edge enhancement.

## **Pseudocode:**

Algorithm ImageEnhancement

Input: Grayscale image f(x,y)

Output: Enhanced images using histogram equalization, spatial filters, and frequency domain filters

Step 1: Read the grayscale image

f ← ReadImage("input\_image.jpg")

Step 2: Plot original histogram

hist\_f ← ComputeHistogram(f)

Plot(hist\_f, title="Original Histogram")

Step 3: Perform histogram equalization f\_eq ← HistogramEqualization(f) hist\_eq ← ComputeHistogram(f\_eq)

Plot(hist\_eq, title="Equalized Histogram")

Step 4: Comment on results

Print("Histogram equalization redistributes pixel intensities to enhance contrast")

Step 5: Implement Spatial Domain Filtering

// Smoothing filters

f\_mean ← ApplyFilter(f, MeanFilter(3x3)) // Average ﬁlter

f\_weighted ← ApplyFilter(f, WeightedAverageFilter()) // e.g., Gaussian weights f\_median ← ApplyFilter(f, MedianFilter(3x3)) // Non-linear, removes salt & pepper

noise

f\_gaussian ← ApplyFilter(f, GaussianFilter(σ=1.0)) // Smooths while preserving edges better than mean

// Display results

Display(f\_mean, "Mean Filtered Image") Display(f\_weighted, "Weighted Average Filtered Image") Display(f\_median, "Median Filtered Image") Display(f\_gaussian, "Gaussian Filtered Image")

Step 6: Implement Frequency Domain Filtering

// Compute Fourier Transform

F ← DFT(f)

// Apply frequency domain filters

F\_lowpass ← ApplyLowPassFilter(F, cutoff\_radius) F\_highpass ← ApplyHighPassFilter(F, cutoff\_radius)

// Inverse transform to get spatial images f\_lowpass ← IDFT(F\_lowpass) f\_highpass ← IDFT(F\_highpass)

Display(f\_lowpass, "Low-Pass Filtered Image") Display(f\_highpass, "High-Pass Filtered Image")

Step 7: Comment on overall results

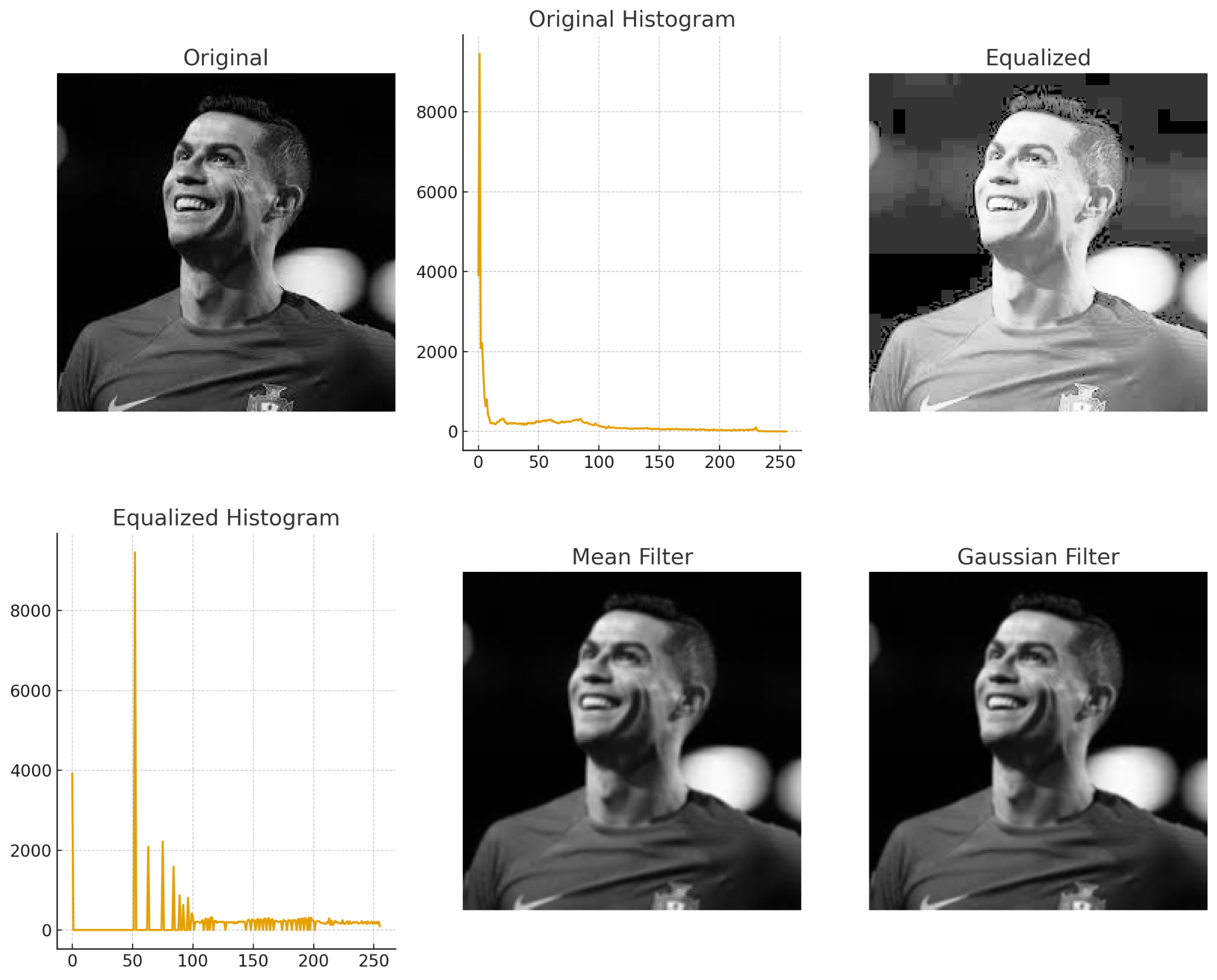
Print("Spatial filters modify local pixel neighborhoods, useful for smoothing, sharpening, and edge detection.")

Print("Frequency domain filters selectively enhance or suppress certain frequency components.")

Print("Histogram equalization enhances overall contrast by redistributing intensity values.")

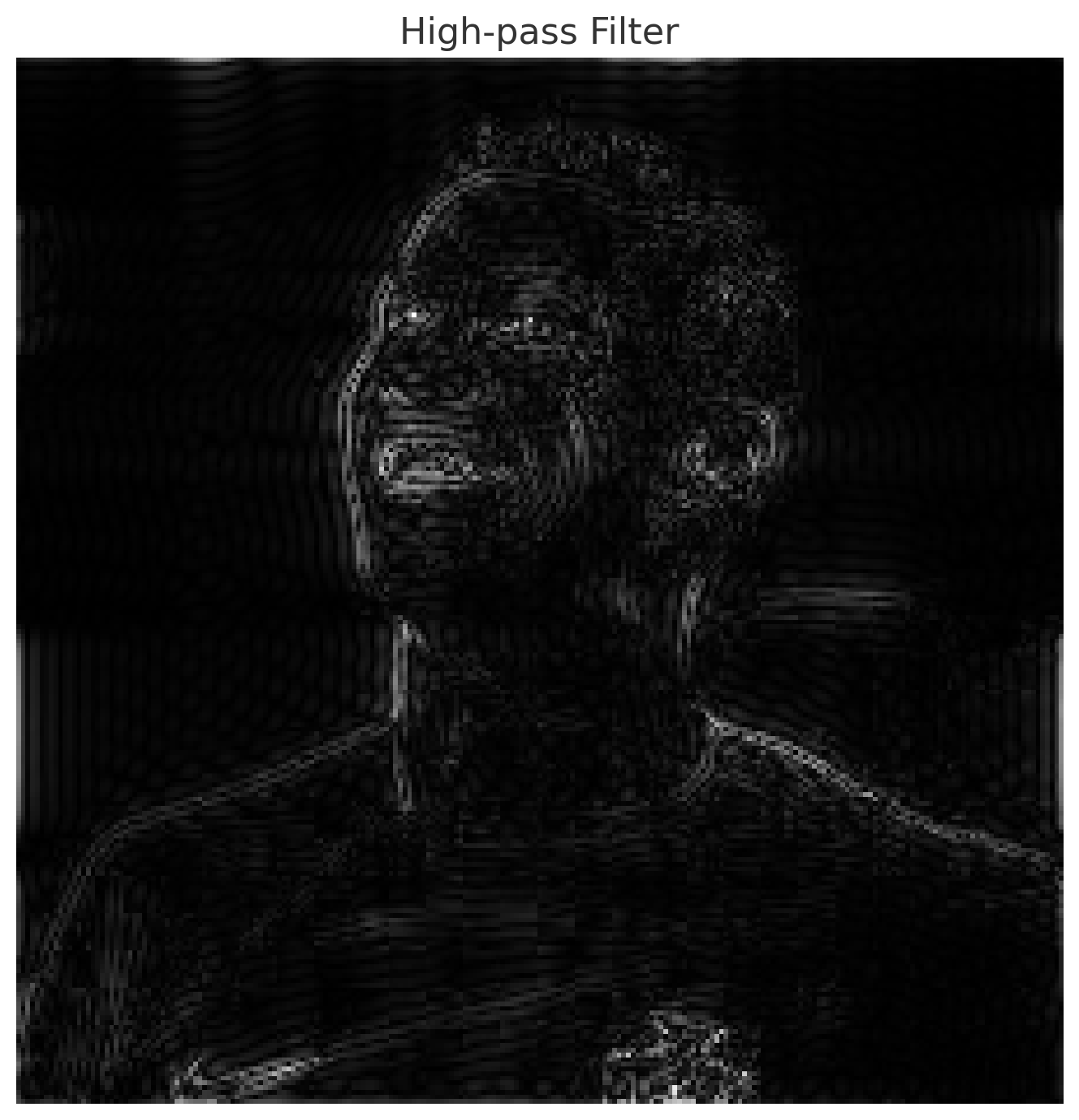
End Algorithm

## **Result:**



Spatial Domain Filter





Frequency Domain Filter

### 

* **Observation:** Histogram equalization redistributes the pixel intensities, which improves the contrast of the image. In the original image, most pixel values were concentrated in darker ranges, but after equalization the histogram becomes more balanced and spread out, making details in the face and background clearer. Spatial domain filters directly operate on pixels: the mean filter smooths the image but blurs edges, the Gaussian filter smooths with less distortion, the median filter removes salt-and-pepper noise while keeping edges sharp, and the Laplacian filter highlights edges and fine details for sharpening. Frequency domain filters work on the image in the Fourier domain: the low-pass filter retains smooth regions while removing fine details and noise (causing blur), whereas the high-pass filter extracts edges and fine structures, making it useful for edge detection.
* **Conclusion:** Histogram equalization is effective for contrast improvement, spatial domain filters are useful for noise reduction and detail enhancement, while frequency domain filters are effective for global smoothing and edge extraction. Together, these methods provide a powerful set of tools for image enhancement in real-world applications such as medical imaging, object recognition, and computer vision.