Name	Adwait Purao
UID no.	2021300101

Experiment 7	
AIM :	Image Enhancement using Spatial Filtering
OBJECTIVE:	● Develop an adaptive enhancement technique capable of effectively regulating illumination and reflectance components in challenging lighting conditions. ● Implement an "Iterative Homomorphic Filter" to iteratively attenuate the illumination component and enhance the reflectance component based on image types. ● Evaluate the proposed filter's performance compared to conventional enhancement schemes, particularly in scenarios with optical sources where conventional methods fail due to excessive amplification and saturation of high-intensity regions. ● Assess the enhancement efficiency of the proposed technique through both subjective and objective measures, aiming to demonstrate its superiority over existing algorithms available in the literature.
INTRODUCTION:	Conventional image enhancement methods often excel in improving dark images but struggle with those containing optical sources, leading to loss of vital original information due to over-amplification and saturation. To address this, an adaptive "Iterative Homomorphic Filter" is proposed. This filter, tailored to different image types, iteratively adjusts illumination and reflectance components based on image intensity. Experimental results show its superiority over existing algorithms in enhancing images under challenging illumination, preserving more original information.

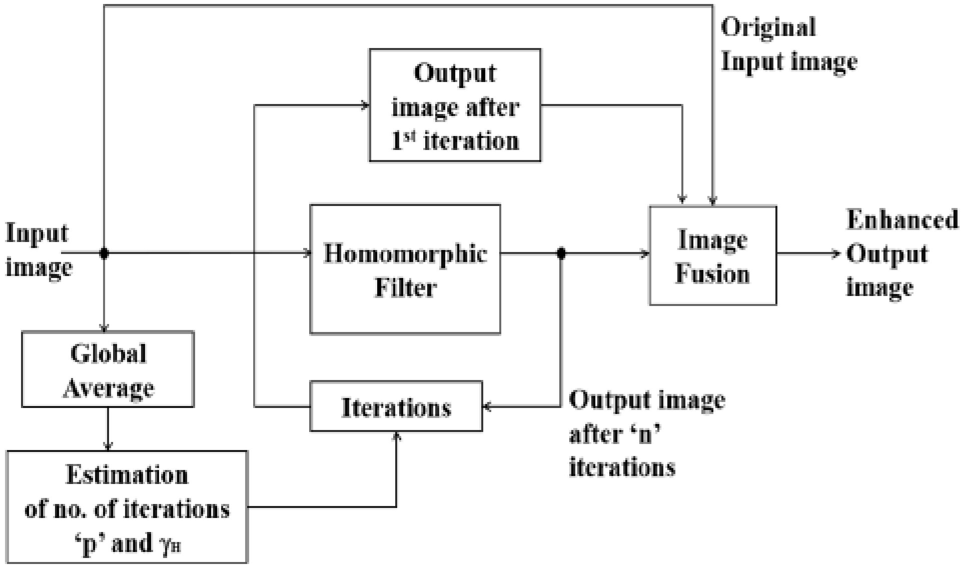
<table>
<tr>
<td><b>BLOCK DIAGRAM:</b></td>
<td>

Original Input image
Output image after 1st iteration
Input image
Homomorphic Filter
Image Fusion
Enhanced Output image
Global Average
Iterations
Output image after 'n' iterations
Estimation of no. of iterations 'p' and γH

Figure 1: Block diagram of proposed Iterative Homomorphic Filtering (IHF)
</td>
</tr>
</table>

B. Flow-Chart of the proposed IHF technique..

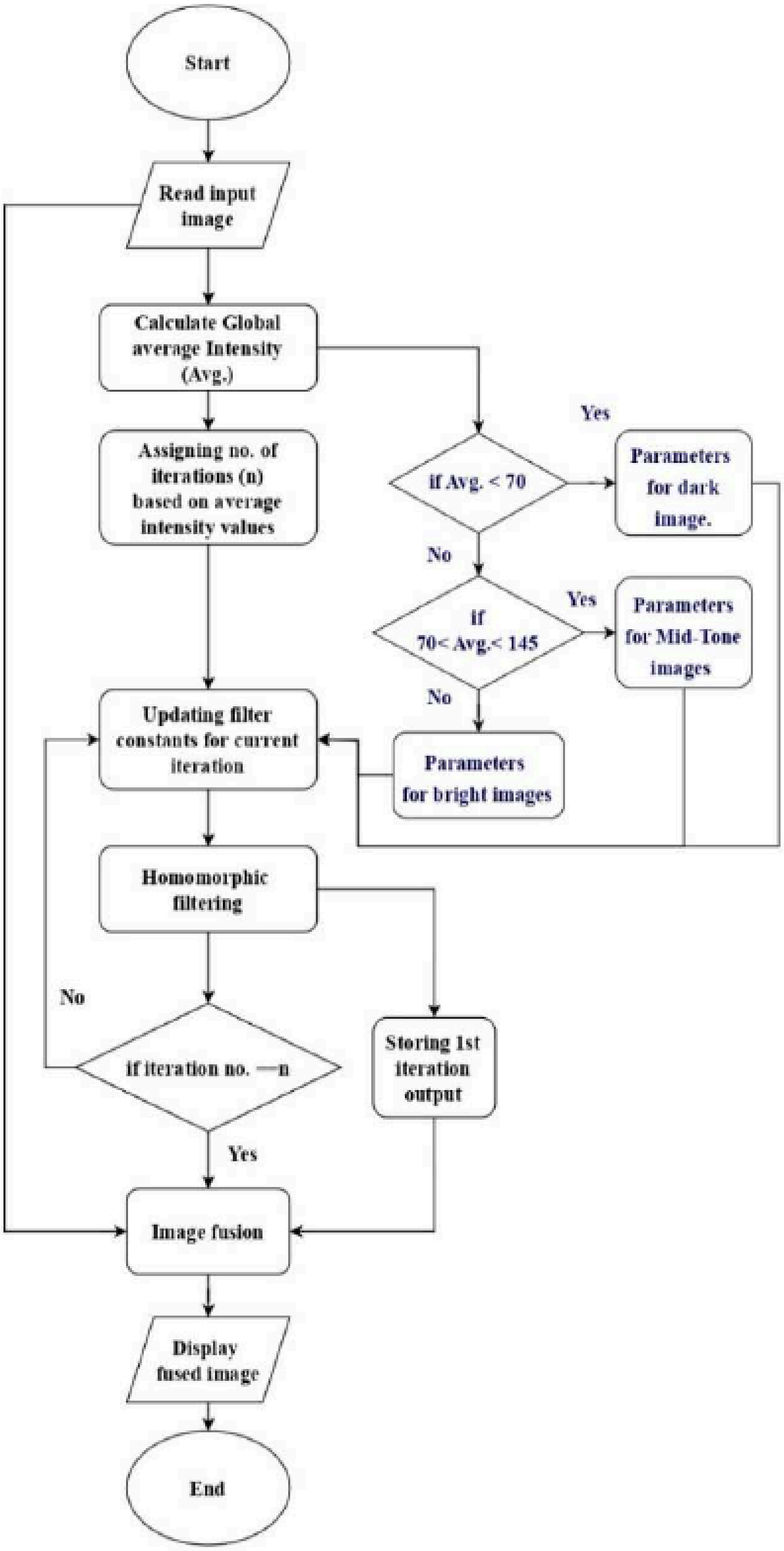

Start
Read input image
Calculate Global average Intensity (Avg.)
Assigning no. of iterations (n) based on average intensity values
if Avg. < 70
Yes
Parameters for dark image.
No
if 70< Avg.< 145
Yes
Parameters for Mid-Tone images
No
Parameters for bright images
Updating filter constants for current iteration
Homomorphic filtering
No
if iteration no. —n
Yes
Storing 1st iteration output
Image fusion
Display fused image
End

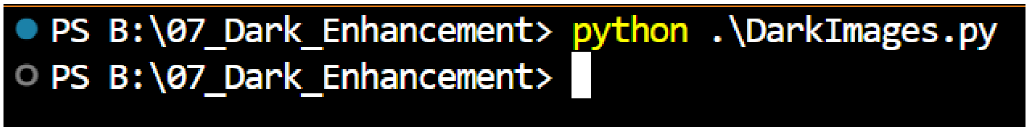
Figure 2: Flow-Chart illustrating Proposed Technique

IMPLEMENTATION:

```python
import numpy as np
import cv2
import matplotlib.pyplot as plt
from scipy.fftpack import fft2, ifft2, fftshift, ifftshift


def global_avg_intensity(img):
    """Calculates the global average intensity of the input image"""
    return np.mean(img)


def allocate_filter_params(avg_intensity):
    """Allocates the IHF filter parameters based on the global average intensity"""
    if avg_intensity < 70:
        h_gain = 2.53
        l_gain = 0.9
        num_iter = 2
        cutoff = 120
    elif 70 <= avg_intensity < 145:
        h_gain = 1.58
        l_gain = 0.9
        num_iter = 3
        cutoff = 1500
    else:
        h_gain = 1.38
        l_gain = 0.9
        num_iter = 3
        cutoff = 1500
    return h_gain, l_gain, num_iter, cutoff


def homomorphic_filter(img, h_gain, l_gain, cutoff):
    """Applies the Homomorphic Filter to the input image"""
    img_log = np.log1p(img)
    img_fft = fftshift(fft2(img_log))
    rows, cols = img.shape
    crow, ccol = rows // 2, cols // 2
    mask = np.zeros((rows, cols), np.float32)
    y, x = np.ogrid[:rows, :cols]
    d = np.sqrt((x - ccol)**2 + (y - crow)**2)
    mask = 1 - np.exp(-(d ** 2) / (2 * (cutoff ** 2)))
    img_filtered = mask * img_fft
    img_ifft = np.real(ifft2(ifftshift(img_filtered)))
    img_enhanced = np.expm1(img_ifft)
    return img_enhanced


def iterative_homomorphic_filtering(img, num_iter):
    """Applies the Iterative Homomorphic Filtering"""
    avg_intensity = global_avg_intensity(img)
    h_gain, l_gain, num_iter, cutoff =
allocate_filter_params(avg_intensity)
    out_1st_iter = homomorphic_filter(img, h_gain, l_gain, cutoff)
    out_final_iter = out_1st_iter.copy()
    for i in range(1, num_iter):
        h_gain = h_gain * np.exp(-0.1 * i)
        l_gain = l_gain * np.exp(-0.1 * i)
        out_final_iter = homomorphic_filter(out_final_iter, h_gain, l_gain, cutoff)

    fused_img = 0.33 * img + 0.33 * out_1st_iter + 0.33 * out_final_iter
    return fused_img

# Load the input image
input_image = cv2.imread('B:\\07_Dark_Enhancement\\small_baby.jpeg', cv2.IMREAD_GRAYSCALE)

# Function to display image and its histogram
def display_with_histogram(title, img):
    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plt.title(title)
    plt.imshow(img, cmap='gray')
    plt.axis('off')

    plt.subplot(1, 2, 2)
    plt.title('Histogram')
    plt.hist(img.ravel(), bins=256, range=(0, 256), color='black',
alpha=0.6)
    plt.xlabel('Pixel Value')
    plt.ylabel('Frequency')

    plt.tight_layout()
    plt.show()

# Display the input image with its histogram
display_with_histogram('Input Image', input_image)

# Apply Histogram Equalization
he_image = cv2.equalizeHist(input_image)
display_with_histogram('Histogram Equalization', he_image)

# Apply Gamma Correction
gc_image = np.uint8(np.power(input_image / 255.0, 0.5) * 255)
display_with_histogram('Gamma Correction', gc_image)

# Apply Local Adaptive Gamma Correction with epsilon to prevent divide by zero
epsilon = 1e-8  # Small value to avoid division by zero
lagc_image = input_image.copy()
for i in range(input_image.shape[0]):
    for j in range(input_image.shape[1]):
        local_mean = np.mean(input_image[max(0, i-10):min(input_image.shape[0], i+10),
                                          max(0,
j-10):min(input_image.shape[1], j+10)]) + epsilon

        gamma = 1 / (local_mean / 255.0)

        lagc_image[i, j] = np.uint8(np.power(input_image[i, j] /
255.0, gamma) * 255)

display_with_histogram('Local Adaptive Gamma Correction',
lagc_image)


# Apply Piecewise Linear Transformation

plt_image = cv2.normalize(input_image, None, 0, 255,
cv2.NORM_MINMAX, cv2.CV_8U)

display_with_histogram('Piecewise Linear Transformation',
plt_image)


# Apply Iterative Homomorphic Filtering

ihf_1st_iter = homomorphic_filter(input_image, 1.58, 0.9, 1500)

ihf_2nd_iter = homomorphic_filter(ihf_1st_iter, 1.58 *
np.exp(-0.1), 0.9 * np.exp(-0.1), 1500)

ihf_fused = 0.33 * input_image + 0.33 * ihf_1st_iter + 0.33 *
ihf_2nd_iter

display_with_histogram('IHF Fused', ihf_fused)
```

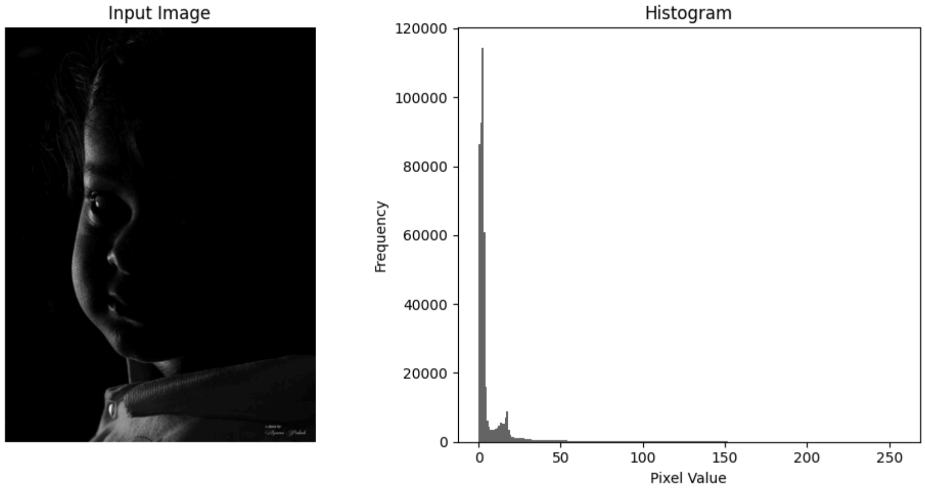
OUTPUT:

Terminal:

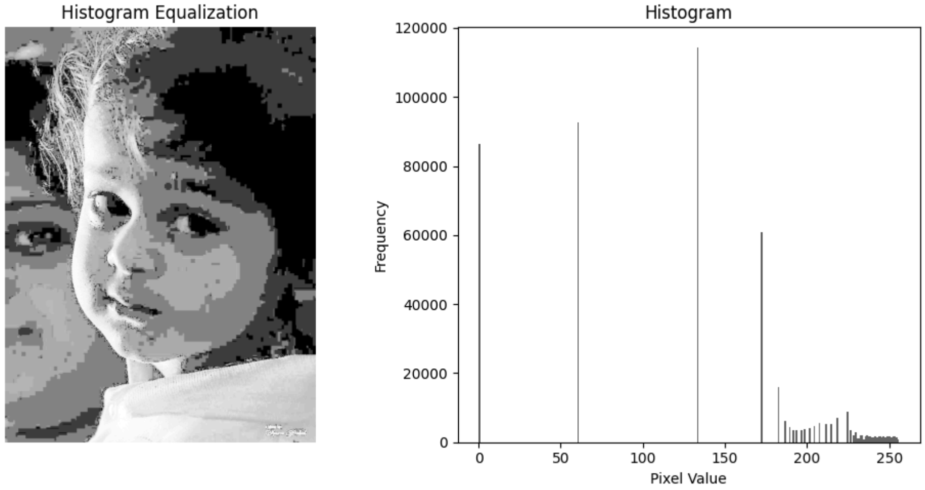


```
● PS B:\07_Dark_Enhancement> python .\DarkImages.py
○ PS B:\07_Dark_Enhancement> █
```

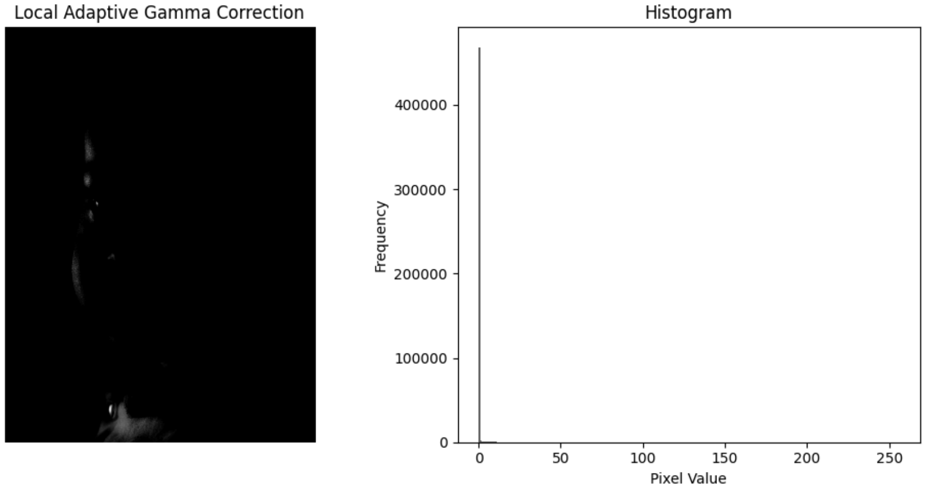
Input Image:



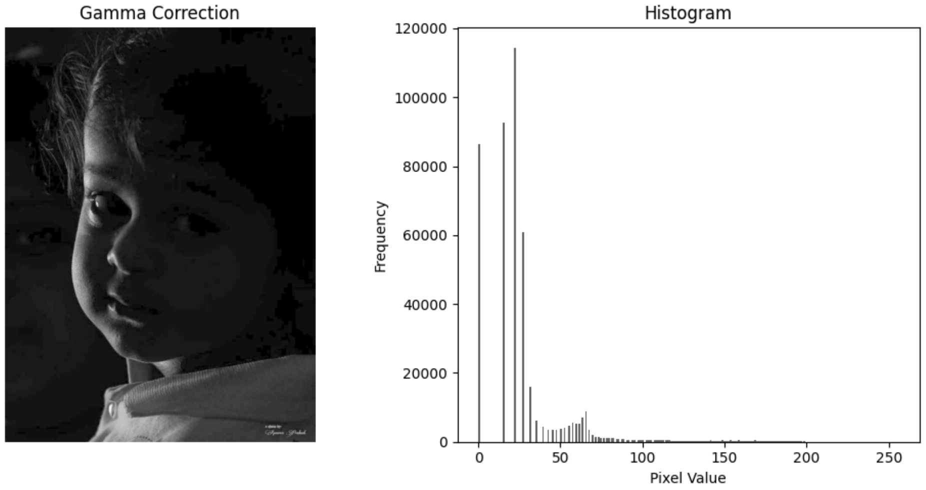
Histogram Equalization:



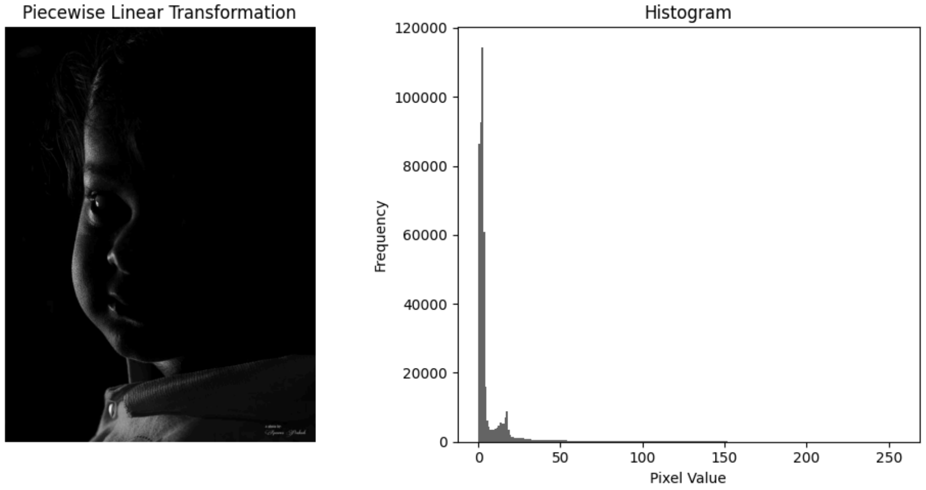
Local Adaptive Gamma Correction :



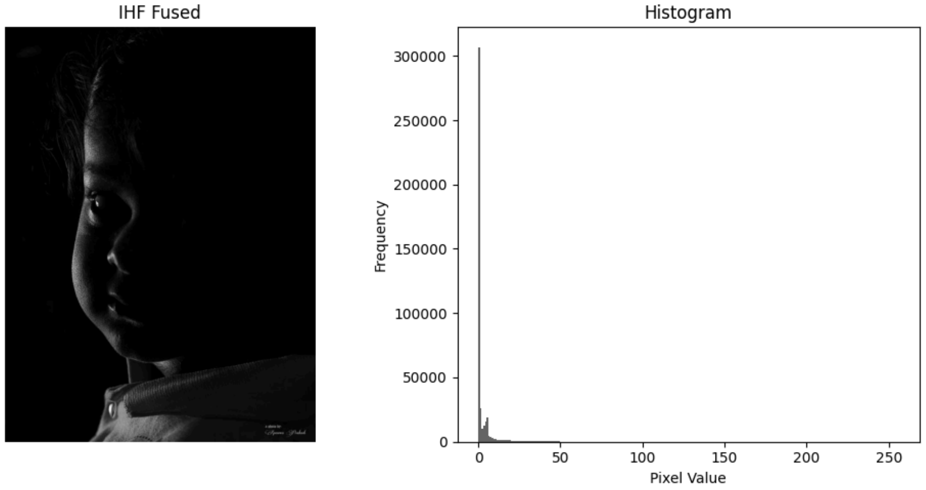
Gamma Correction:



Piecewise Linear Transformation:



IHF Fused:



<table>
<tr><td></td><td></td></tr>
<tr><td><b>REFERENCE:</b></td><td>A. Nayak and A. Acharya, "Enhancement of Dark Images in Presence of Optical Sources Using Iterative Homomorphic Filter (IHF)," 2023 1st International Conference on Circuits, Power and Intelligent Systems (CCPIS), Bhubaneswar, India, 2023, pp. 1-7, doi: 10.1109/CCPIS59145.2023.10291651. keywords: {Optical filters;Reflectivity;Optical attenuators;Lighting;Optical saturation;Filtering algorithms;Traffic control;dark image enhancement;homomorphic filtering;contrast improvement;image fusion},

https://ieeexplore.ieee.org/document/10291651</td></tr>
<tr><td colspan="2"><b>CONCLUSION:</b>
In conclusion, the proposed "Iterative Homomorphic Filter" presents a promising solution to the limitations of conventional image enhancement techniques, particularly in handling images with optical sources. By effectively regulating illumination and reflectance components based on image types, this adaptive filter demonstrates superior performance in preserving original information while enhancing image quality. The experimental results underscore its efficacy, surpassing existing algorithms in achieving enhanced images under challenging illumination conditions. This signifies its potential for practical applications in various fields requiring image enhancement, promising better preservation of crucial details in visually demanding scenarios.</td></tr>
</table>