B_A3_Feature_Extraction

In [1]:
```
# Import necessary libraries
from skimage.io import imread  # For reading images
from skimage.color import rgb2gray  # For converting color images to grayscale
from skimage.filters import threshold_otsu, gaussian  # For Otsu's thresholding and Gaus
import matplotlib.pyplot as plt  # For displaying images and plotting
import numpy as np  # For numerical operations
```

In [2]:
```
# Function to display a single image
def show_img(image, title="Image"):
    plt.imshow(image)
    plt.title(title)
    plt.axis('off')  # Hide axes for better display
    plt.show()
```

In [3]:
```
# Function to display multiple images
def show_images(images, titles):
    n = len(images)
    fig, axes = plt.subplots(1, n, figsize=(15, 5))  # Create subplots for each image
    for i in range(n):
        ax = axes[i]
        ax.imshow(images[i], cmap='gray' if len(images[i].shape) == 2 else None)
        ax.set_title(titles[i])
        ax.axis('off')  # Hide axes
    plt.show()
```

In [4]:
```
# Load the image
image = imread(r"C:\\Users\\asus\\Downloads\\suhail\\ISR LAb\\123.jpg")
```

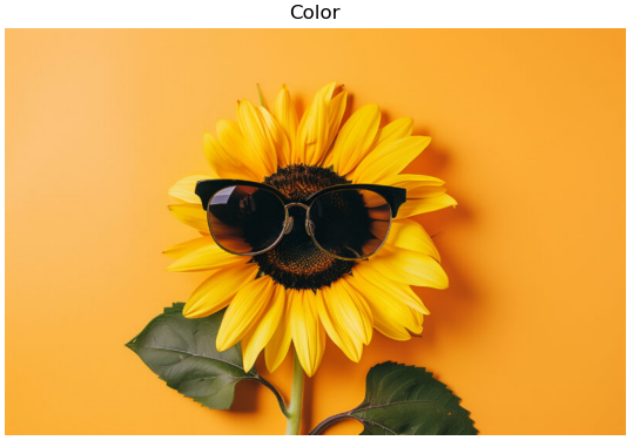
In [5]:
```
# Display the original image
show_img(image, "Original Image")
```

In [6]:
```
# Create red and yellow channel images
red, yellow = image.copy(), image.copy()

# Remove green and blue channels for the red image
red[:, :, (1, 2)] = 0  # Set green and blue channels to zero

# Remove blue channel for the yellow image
yellow[:, :, 2] = 0  # Set the blue channel to zero
```

In [7]:
```
# Show the red and yellow intensity images
show_images(images=[red, yellow], titles=['Red Intensity', 'Yellow Intensity'])
```

In [8]:
```
# Convert the original image to grayscale
gray_image = rgb2gray(image)
```

In [9]:
```
# Show the color and grayscale images
show_images(images=[image, gray_image], titles=["Color", "Grayscale"])
```

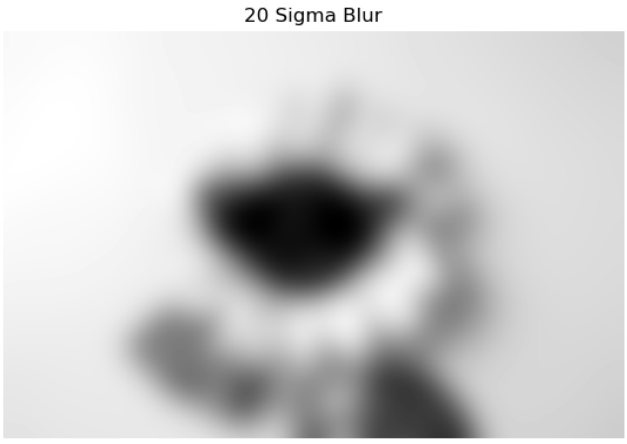


In [10]:
```
# Print image shape information
print("Colored image shape:", image.shape)
print("Grayscale image shape:", gray_image.shape)
```

```
Colored image shape: (525, 800, 3)
Grayscale image shape: (525, 800)
```

In [12]:
```
# Apply Otsu's thresholding
thresh = threshold_otsu(gray_image)

# Create a binary image based on the threshold
binary_image = gray_image > thresh

# Display grayscale and Otsu binary images
show_images(images=[gray_image, binary_image], titles=["Grayscale", "Otsu Binary"])
```

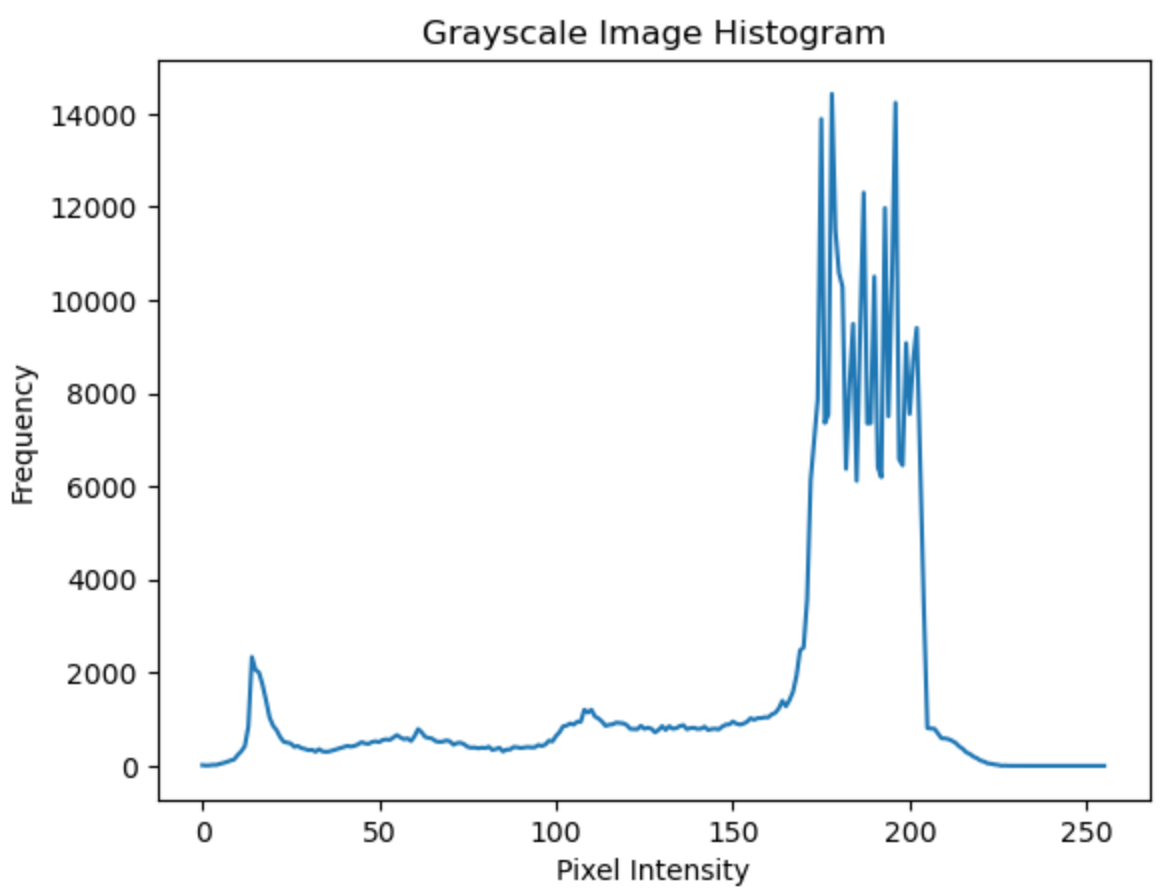
In [13]:
```
# Apply Gaussian filter (blurring) to the grayscale image
blurred_image = gaussian(gray_image, sigma=20)

# Show the grayscale and blurred images
show_images(images=[gray_image, blurred_image], titles=["Gray Image", "20 Sigma Blur"])
```



In [14]:
```
# Importing matplotlib for image processing
import matplotlib.pyplot as plt

# Load an image in PNG format
img = plt.imread('C:\\Users\\asus\\Downloads\\suhail\\ISR LAb\\123.jpg')

# Display the loaded PNG image
plt.imshow(img)
plt.title('Loaded JPG Image')
plt.axis('off')  # Hide axes
plt.show()
```

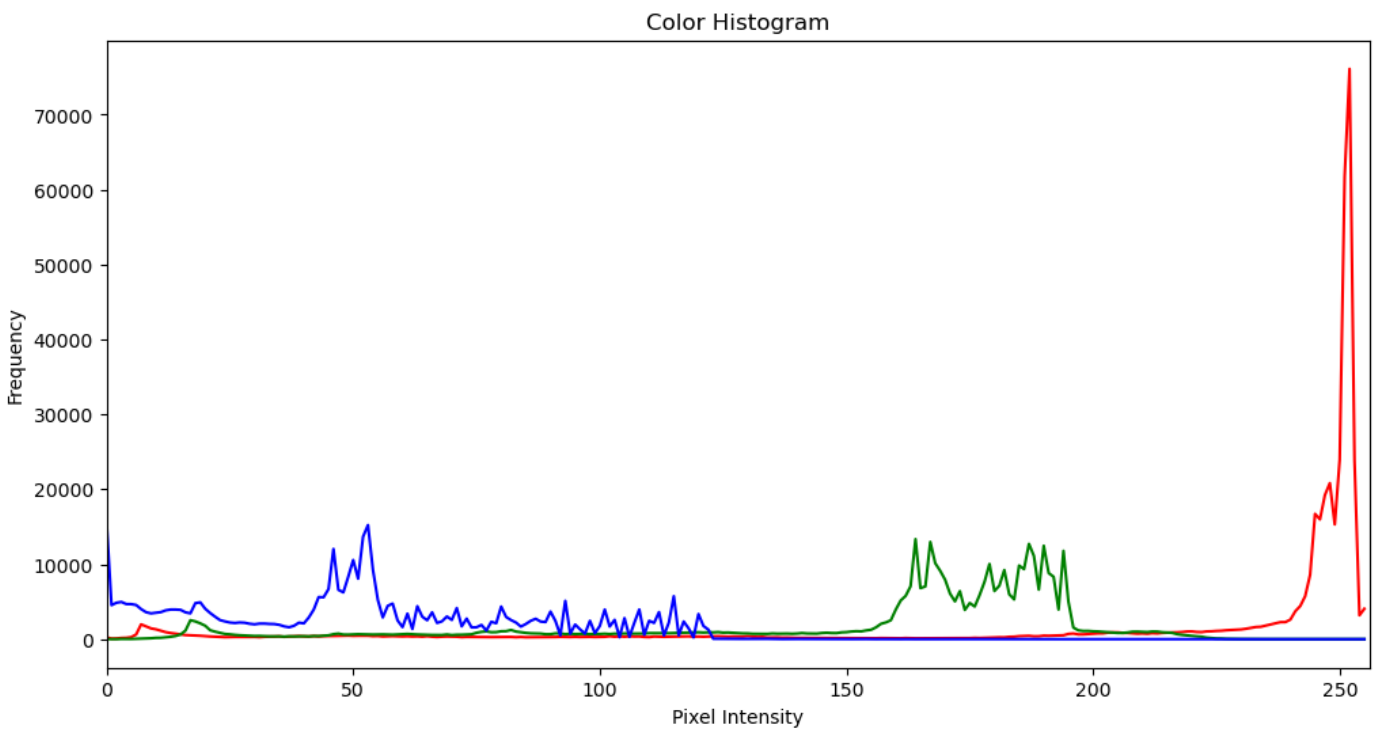
In [15]:

```python
# Example: Using OpenCV to read and analyze an image
import cv2

# Load an image using OpenCV in grayscale mode
img_cv = cv2.imread('C:\\Users\\asus\\Downloads\\suhail\\ISR LAb\\123.jpg', 0)

# Calculate the frequency of pixel intensities (0-255) for the grayscale image
histg = cv2.calcHist([img_cv], [0], None, [256], [0, 256])

# Plotting the histogram using matplotlib
plt.plot(histg)
plt.title("Grayscale Image Histogram")
plt.xlabel("Pixel Intensity")
plt.ylabel("Frequency")
plt.show()
```



In [1]:

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
from skimage import io

image_path = "123.jpg" # Replace with your image path
image = cv2.imread(image_path)
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

def plot_color_histogram(image):
    color = ('r', 'g', 'b')
    plt.figure(figsize=(12, 6))
    for i, col in enumerate(color):
        hist = cv2.calcHist([image], [i], None, [256], [0, 256])
        plt.plot(hist, color=col)
        plt.xlim([0, 256])
    plt.title('Color Histogram')
    plt.xlabel('Pixel Intensity')
    plt.ylabel('Frequency')
    plt.show()

# Plot color histogram
plot_color_histogram(image_rgb)
```


Color Histogram
Frequency
Pixel Intensity
0
10000
20000
30000
40000
50000
60000
70000
0
50
100
150
200
250