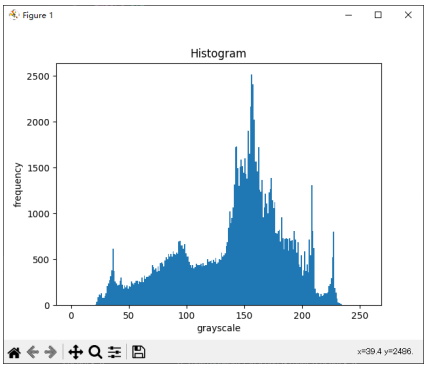
openCV

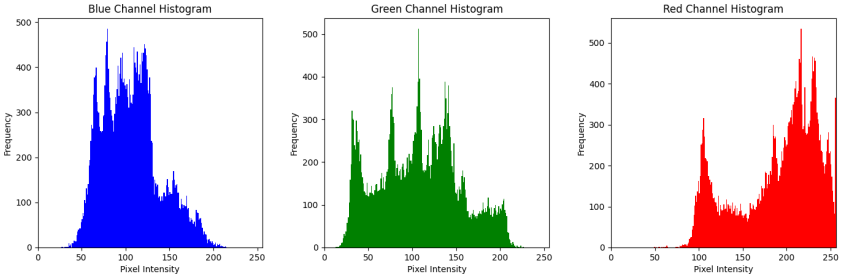
In this lesson, the teacher explained gray histogram extraction, image brightness adjustment, contrast adjustment, color shifting, and both linear and nonlinear changes to images. However, since I didn't have the image resources used in the course, I could only take screenshots during the lesson, resulting in my outcomes differing slightly from those presented in the course.

```
import cv2
from matplotlib import pyplot as plt
image_path = "image/gra.png"
image = cv2.imread(image_path, 0)
plt.hist(image.ravel(), 256, [0, 256])
plt.show()
```

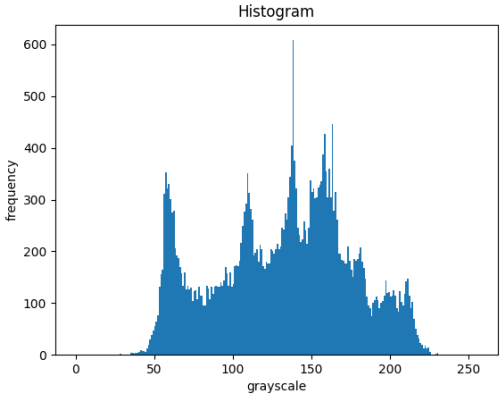



```
image_path2 = "image/gra2.png"
image2 = cv2.imread(image_path2)
channels = cv2.split(image2)
colors = ('b', 'g', 'r')
channel_names = ('Blue', 'Green', 'Red')
plt.figure(figsize=(10, 6))
for i, (channel, color, name) in enumerate(zip(channels, colo
    plt.subplot(1, 3, i+1)
    plt.hist(channel.ravel(), 256, [0, 256], color=color)
```

```
    plt.title(f'{name} Channel Histogram')
    plt.xlim([0, 256])
    plt.xlabel('Pixel Intensity')
    plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```

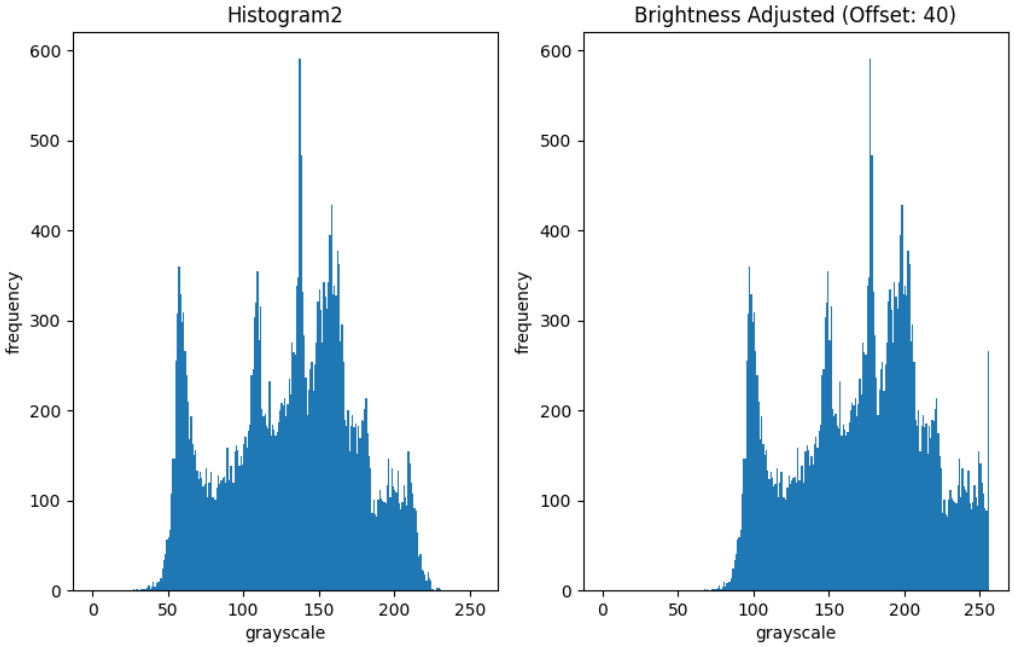


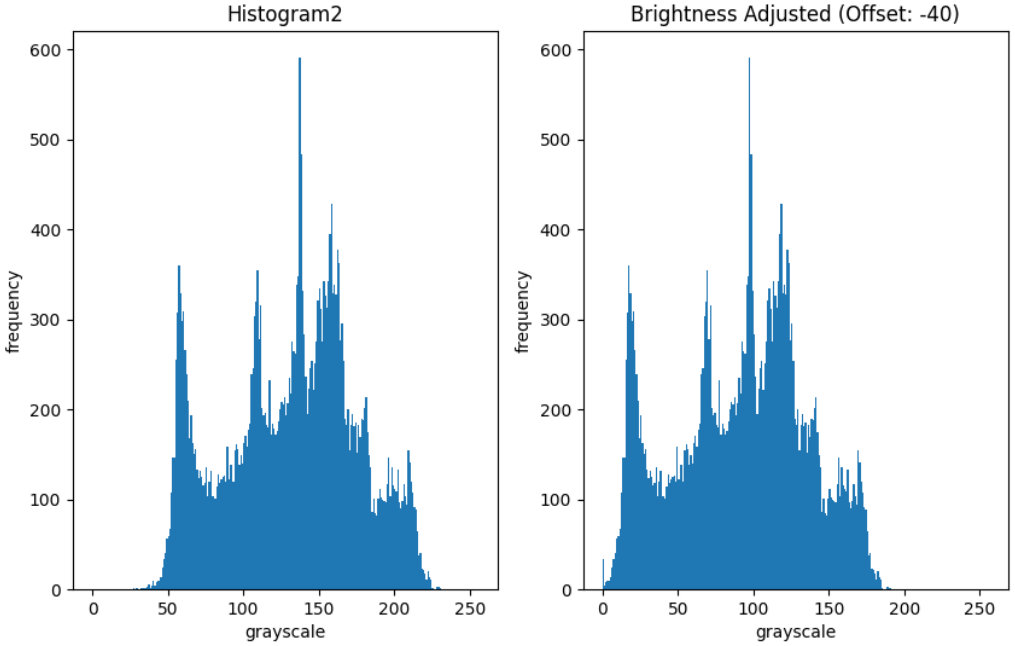

```
image2 = cv2.cvtColor(image2, cv2.COLOR_BGR2GRAY)
plt.hist(image2.ravel(), 256, [0, 256])
plt.title('Histogram')
plt.xlabel('grayscale')
plt.ylabel('frequency')
plt.show()
```




```
image_path2 = "image/gra2.png"
gray_image = cv2.imread(image_path2, 0)
```

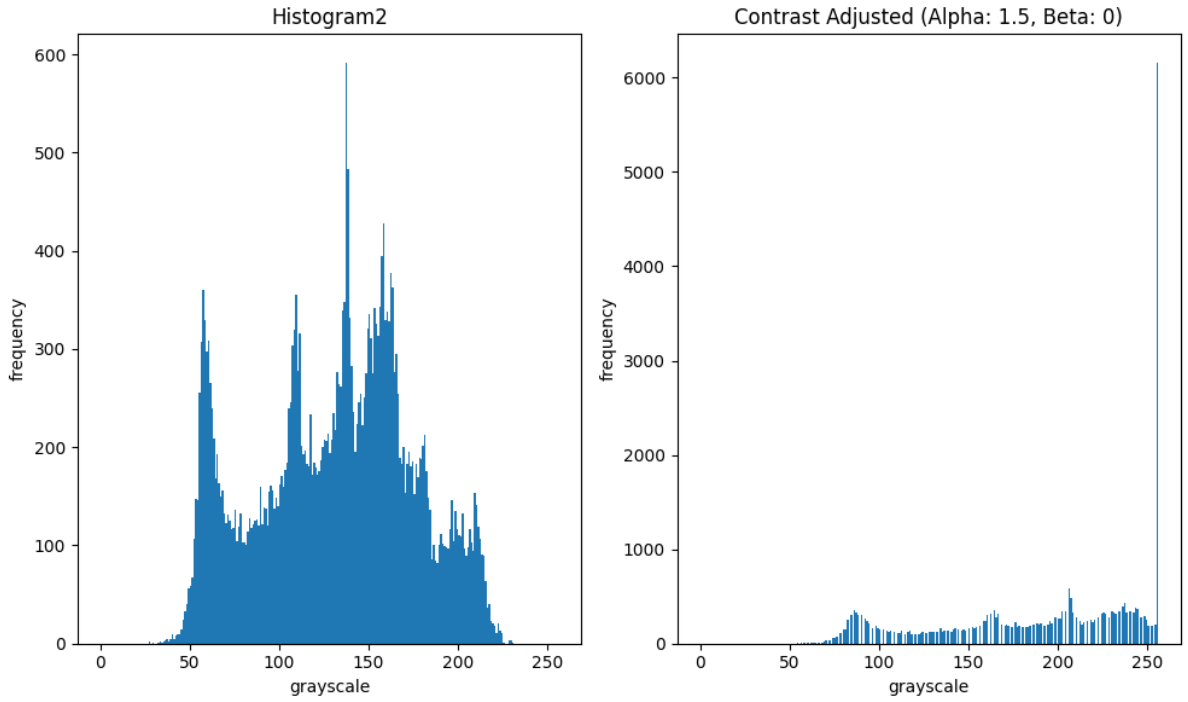
```
plt.figure(figsize=(10, 6))
plt.subplot(1, 2, 1)
plt.hist(gray_image.ravel(), 256, [0, 256])
plt.title('Histogram2')
plt.xlabel('grayscale')
plt.ylabel('frequency')
brightness_offset = 40
bright_image = cv2.add(gray_image, brightness_offset)
bright_image = np.clip(bright_image, 0, 255).astype(np.uint8)
plt.subplot(1, 2, 2)
plt.hist(bright_image.ravel(), 256, [0, 256])
plt.title(f'Brightness Adjusted (Offset: {brightness_offset})
plt.xlabel('grayscale')
plt.ylabel('frequency')
plt.show()
```

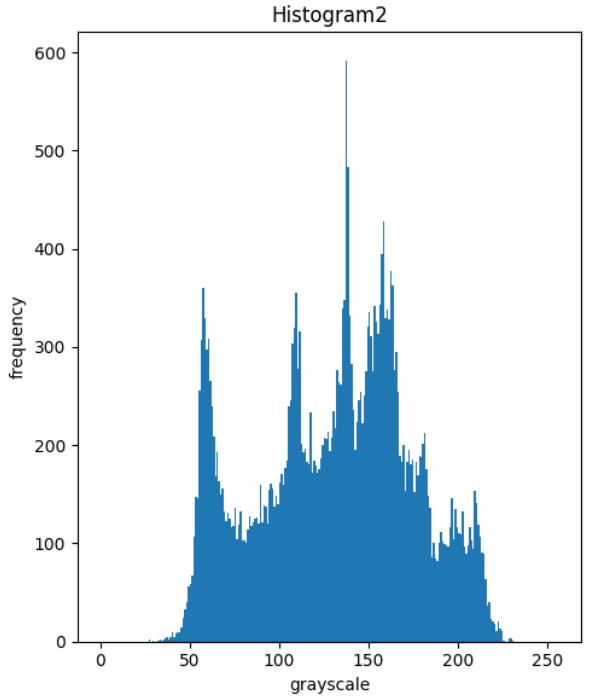


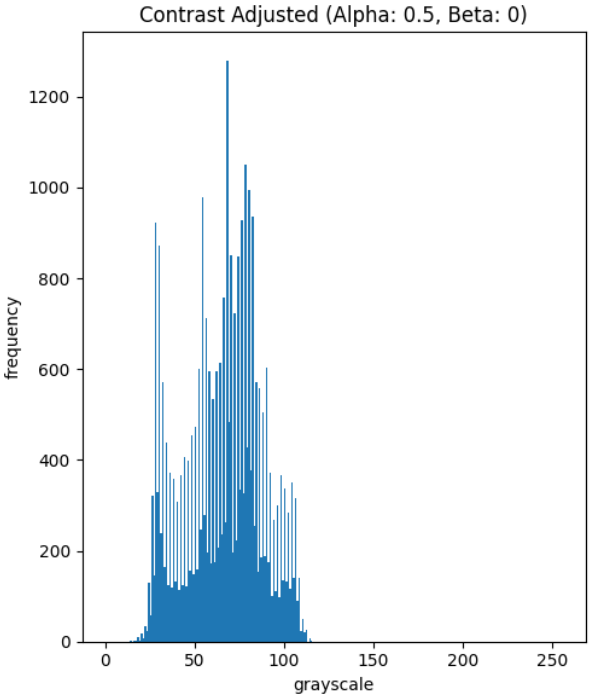



```
alpha = 1.5
beta = 0
# use opencv function
contrast_image = cv2.convertScaleAbs(gray_image, alpha=alpha,

# Adjust the contrast and brightness of an image pixel by pix
```

```
# adjusted_image = alpha * gray_image + beta
# contrast_image = np.clip(adjusted_image, 0, 255).astype(np.
```

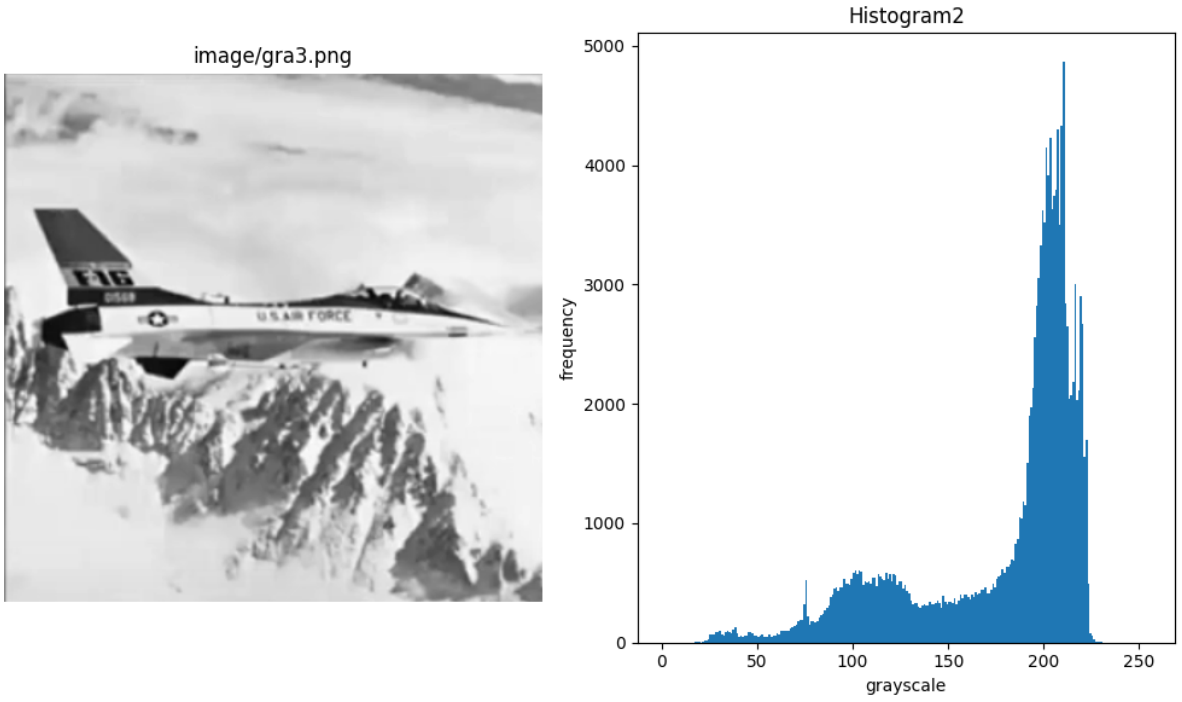




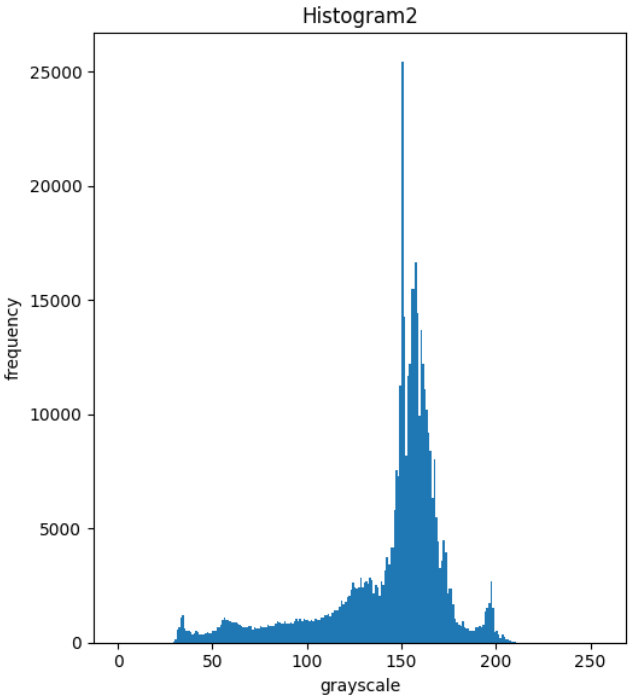



```
import cv2
import numpy as np
```

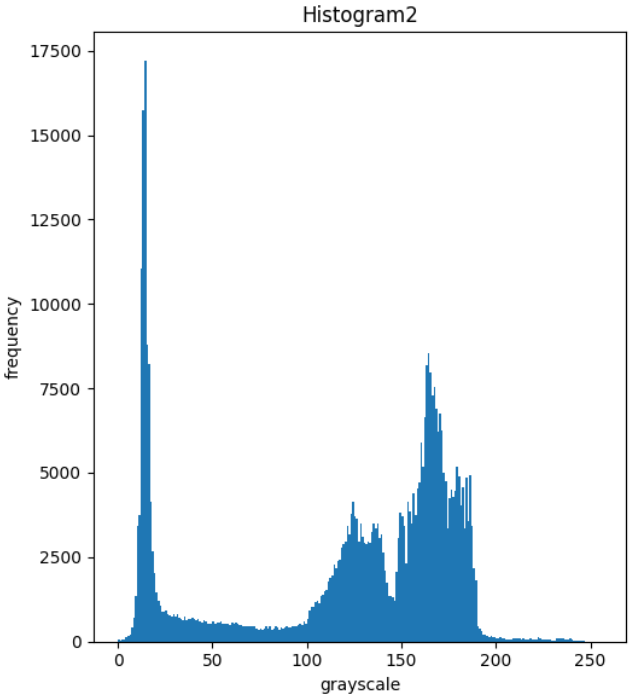
```
from matplotlib import pyplot as plt
image_path = "image/gra3.png"
gray_image = cv2.imread(image_path, 0)
plt.figure(figsize=(10, 6))
plt.subplot(1, 2, 1)
plt.imshow(gray_image, cmap='gray')
plt.title(f'{image_path}')
plt.axis('off')
plt.subplot(1, 2, 2)
plt.hist(gray_image.ravel(), 256, [0, 256])
plt.title('Histogram2')
plt.xlabel('grayscale')
plt.ylabel('frequency')
plt.tight_layout()
plt.show()
```



image/gra4.png

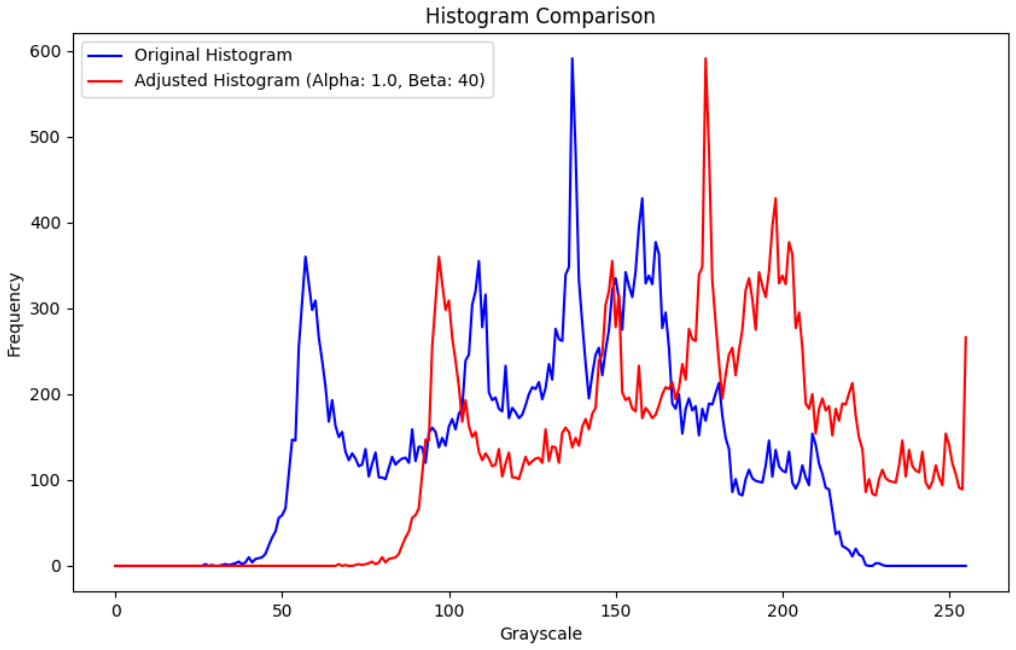


image/gra5.png



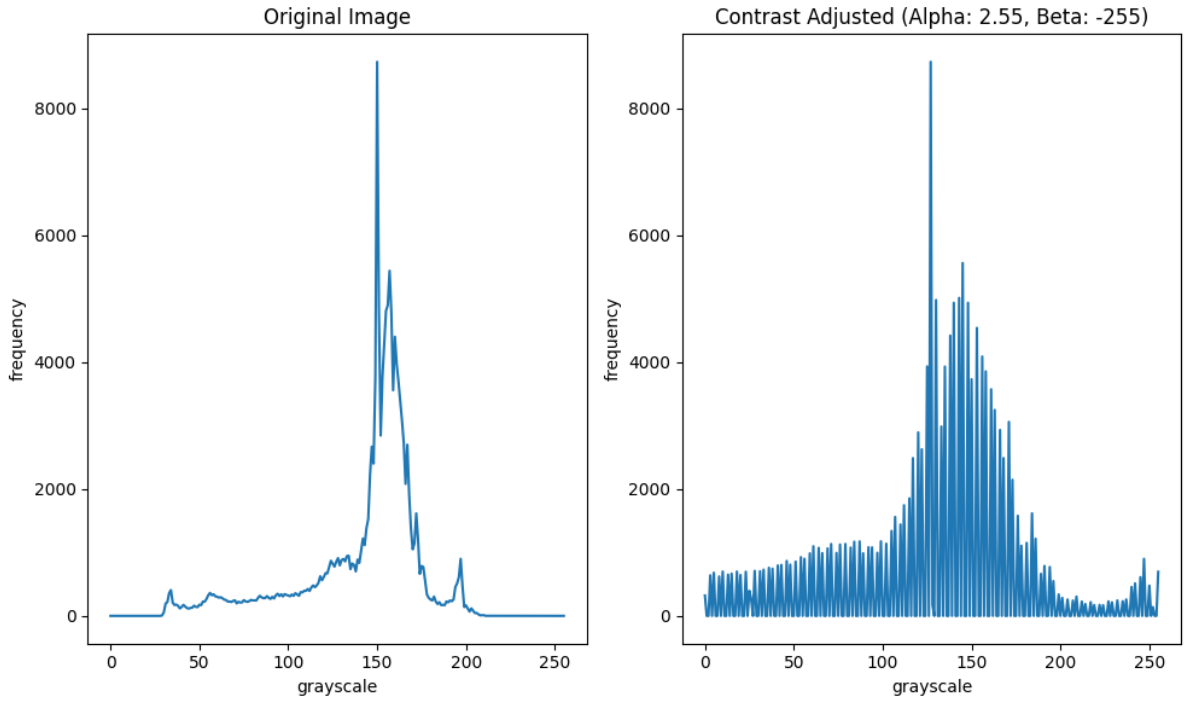

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
image_path2 = "image/gra2.png"
gray_image = cv2.imread(image_path2, cv2.IMREAD_GRAYSCALE)
alpha = 1.0
```

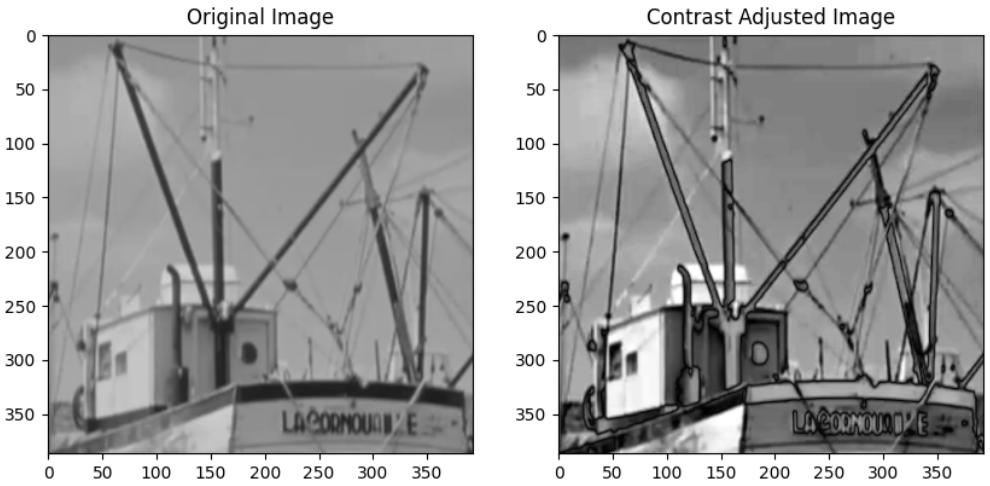
```
beta = 40
adjusted_image = alpha * gray_image + beta
adjusted_image = np.clip(adjusted_image, 0, 255).astype(np.ui
hist_original = cv2.calcHist([gray_image], [0], None, [256],
hist_adjusted = cv2.calcHist([adjusted_image], [0], None, [25
plt.figure(figsize=(10, 6))
plt.plot(hist_original, color='blue', label='Original Histogr
plt.plot(hist_adjusted, color='red',
                                    label=f'Adjusted Histogra
plt.legend()
plt.title('Histogram Comparison')
plt.xlabel('Grayscale')
plt.ylabel('Frequency')
plt.show()
```



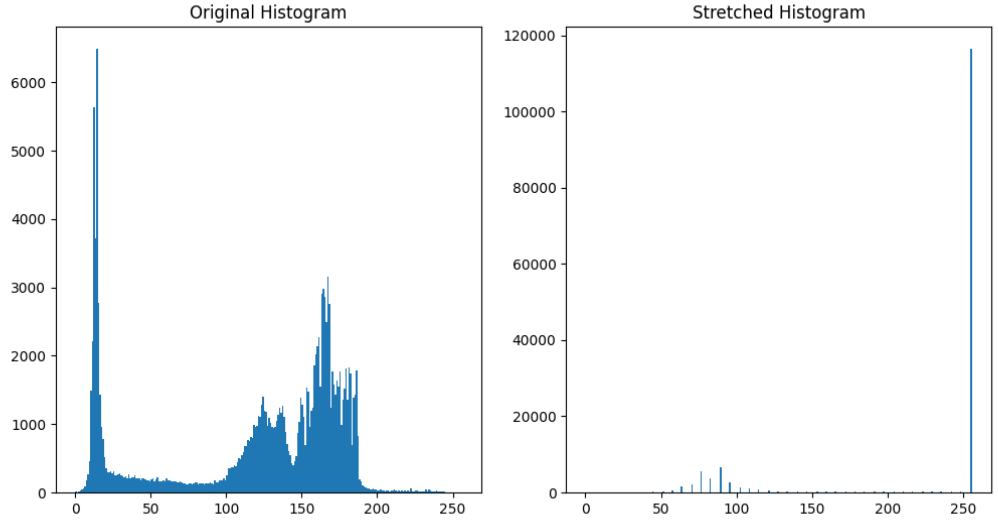

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
image = cv2.imread( "image/gra4.png",0)
```

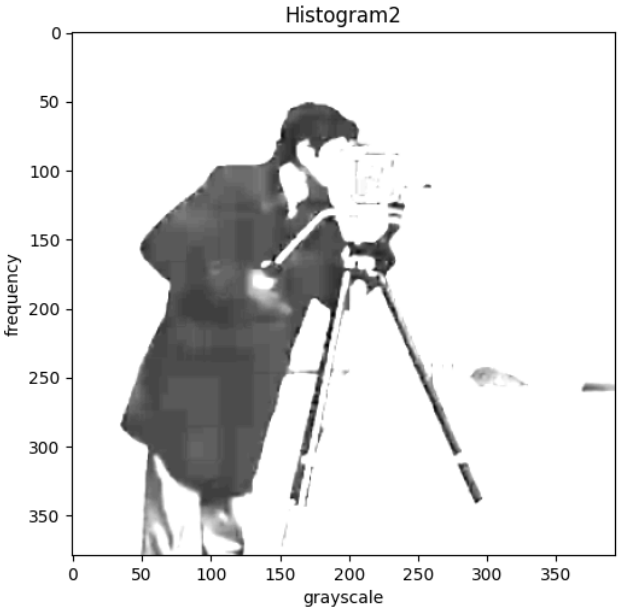
```
alpha = 2.55  # 缩放因子
beta = -255    # 偏移量
transformed_image = cv2.convertScaleAbs(image, alpha=alpha, b
histogram2 = cv2.calcHist([transformed_image], [0], None, [25

plt.figure(figsize=(10, 6))
plt.subplot(1, 2, 1)
histogram = cv2.calcHist([image], [0], None, [256], [0, 256])
plt.plot(histogram)
plt.title(f'Original Image')
plt.xlabel('grayscale')
plt.ylabel('frequency')
plt.subplot(1, 2, 2)
plt.plot(histogram2)
plt.title(f'Contrast Adjusted (Alpha: {alpha}, Beta: {beta})'
plt.xlabel('grayscale')
plt.ylabel('frequency')
plt.tight_layout()
plt.show()
```





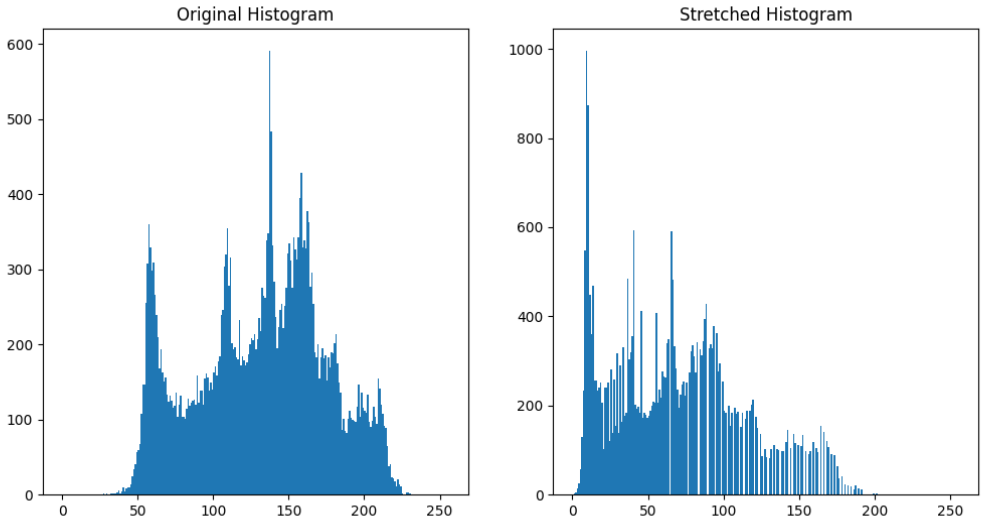

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
img = cv2.imread( "image/gra5.png",0)  # 替换为实际的图像路径


def stretch_histogram_preserve_counts(img, low_in, high_in, l
    normalized = np.clip((img - low_in) / (high_in - low_in),
    stretched_img = normalized * (high_out - low_out) + low_o
    return np.uint8(stretched_img)
# Stretch the values from 0-30 to 0-255
stretched_img = stretch_histogram_preserve_counts(img, 0, 30,
fig, axes = plt.subplots(1, 2, figsize=(12, 6))
axes[0].hist(img.ravel(), bins=256, range=[0, 256])
axes[0].set_title("Original Histogram")
axes[1].hist(stretched_img.ravel(), bins=256, range=[0, 256])
axes[1].set_title("Stretched Histogram")
plt.show()
```

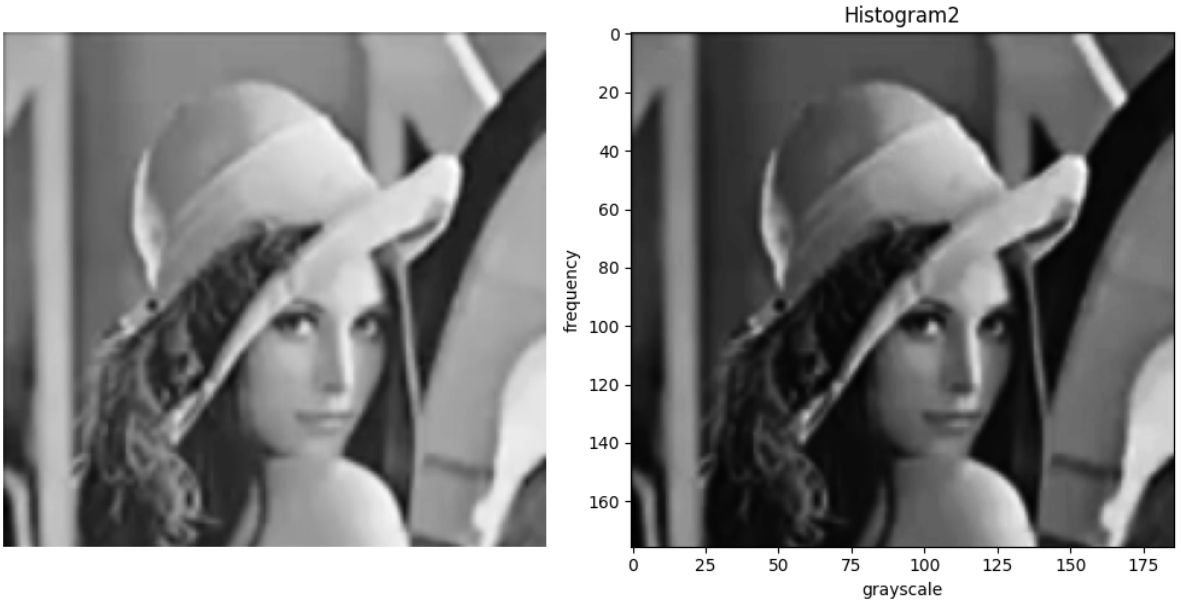




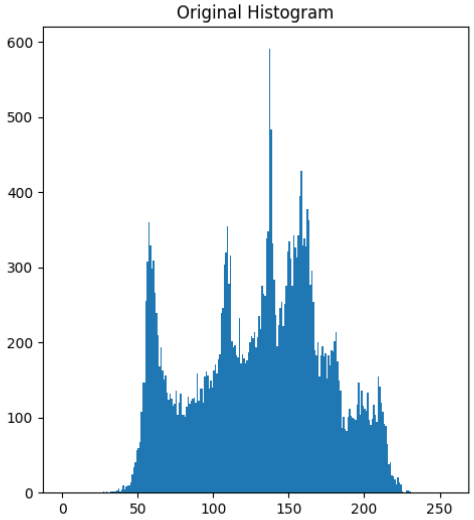

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
img = cv2.imread( "image/gra2.png",0)
def apply_gamma_transformation(img, gamma):
    normalized_img = img / 255.0
    gamma_corrected = np.power(normalized_img, gamma)
    transformed_img = np.uint8(gamma_corrected * 255)
```

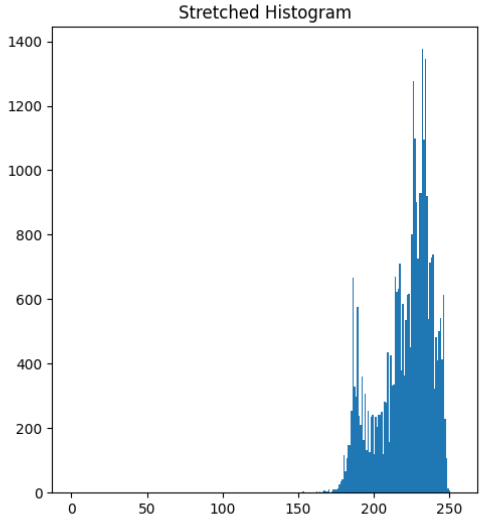
```
    return transformed_img
gamma_value = 2.2
stretched_img =  apply_gamma_transformation(img, gamma_value)
fig, axes = plt.subplots(1, 2, figsize=(12, 6))
axes[0].hist(img.ravel(), bins=256, range=[0, 256])
axes[0].set_title("Original Histogram")
axes[1].hist(stretched_img.ravel(), bins=256, range=[0, 256])
axes[1].set_title("Stretched Histogram")
plt.show()
```

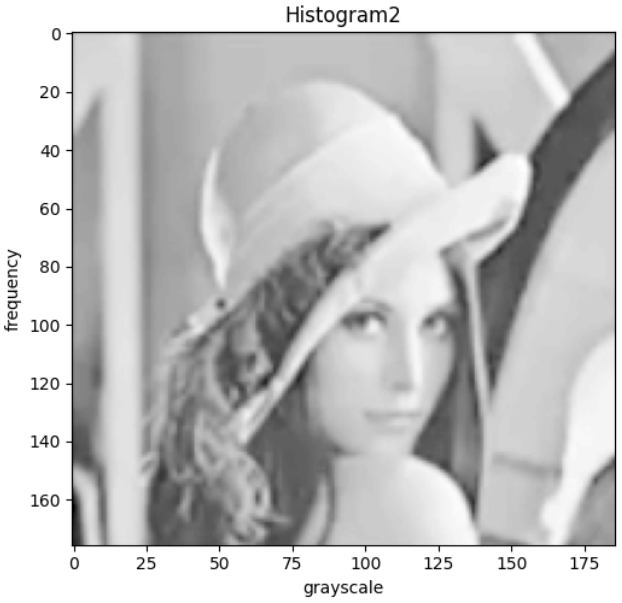





```
import cv2
import numpy as np
from matplotlib import pyplot as plt
img = cv2.imread( "image/gra2.png",0)
def log(c, img):
    c = 255 / np.log(1 + 255)
    output = c * np.log(1 + img)
    output = np.uint8(output)
    return output
stretched_img = log(1, img)
fig, axes = plt.subplots(1, 2, figsize=(12, 6))

axes[0].hist(img.ravel(), bins=256, range=[0, 256])
axes[0].set_title("Original Histogram")
axes[1].hist(stretched_img.ravel(), bins=256, range=[0, 256])
axes[1].set_title("Stretched Histogram")
plt.show()
```

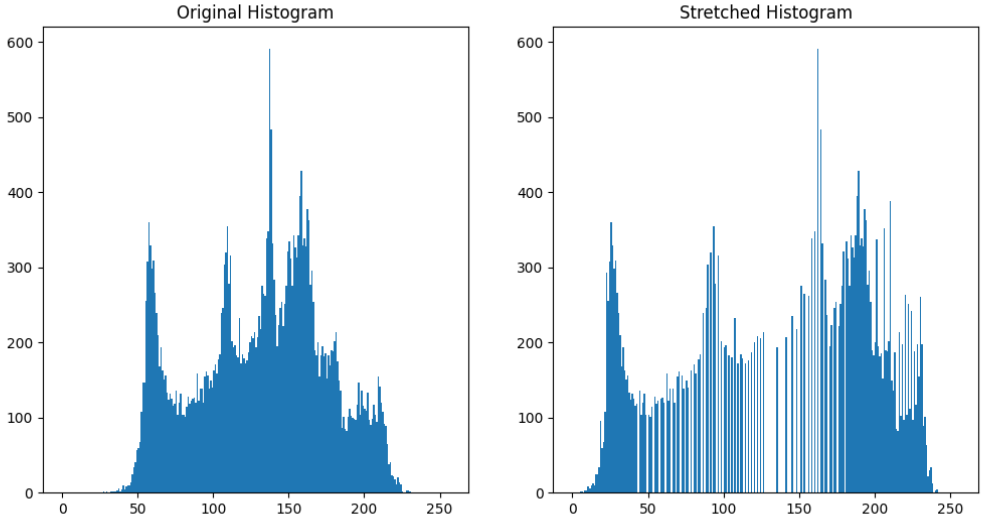


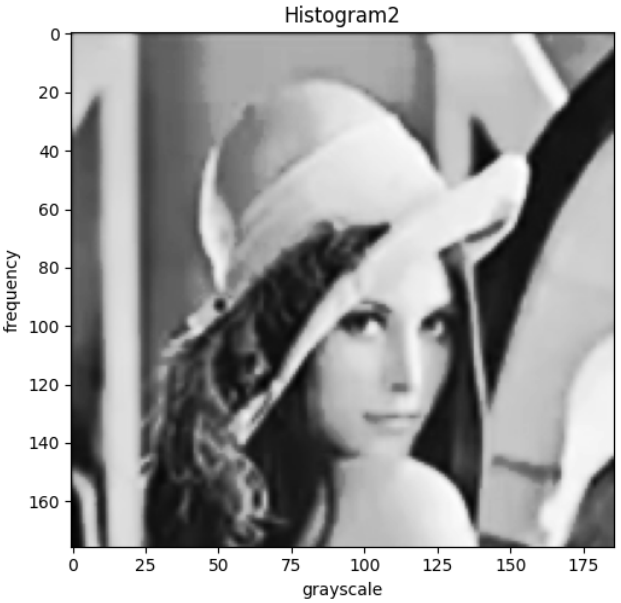





```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread( "image/gra2.png",0)
def log2(img):
    z_m = 255
    z = img
```

```
    z_prime = np.zeros_like(z)
    #  0 <= z <= z_m / 2
    mask1 = z <= z_m / 2
    z_prime[mask1] = 2 * z_m * (z[mask1] / z_m) ** 2
    #  z_m / 2 <= z <= z_m
    mask2 = z > z_m / 2
    z_prime[mask2] = (z_m / 2) * (1 + np.sqrt((2 * z[mask2] -
    return z_prime
stretched_img = log2(img)
fig, axes = plt.subplots(1, 2, figsize=(12, 6))
axes[0].hist(img.ravel(), bins=256, range=[0, 256])
axes[0].set_title("Original Histogram")
axes[1].hist(stretched_img.ravel(), bins=256, range=[0, 256])
axes[1].set_title("Stretched Histogram")
plt.show()
```




Histogram2
frequency
grayscale
0
20
40
60
80
100
120
140
160
0
25
50
75
100
125
150
175