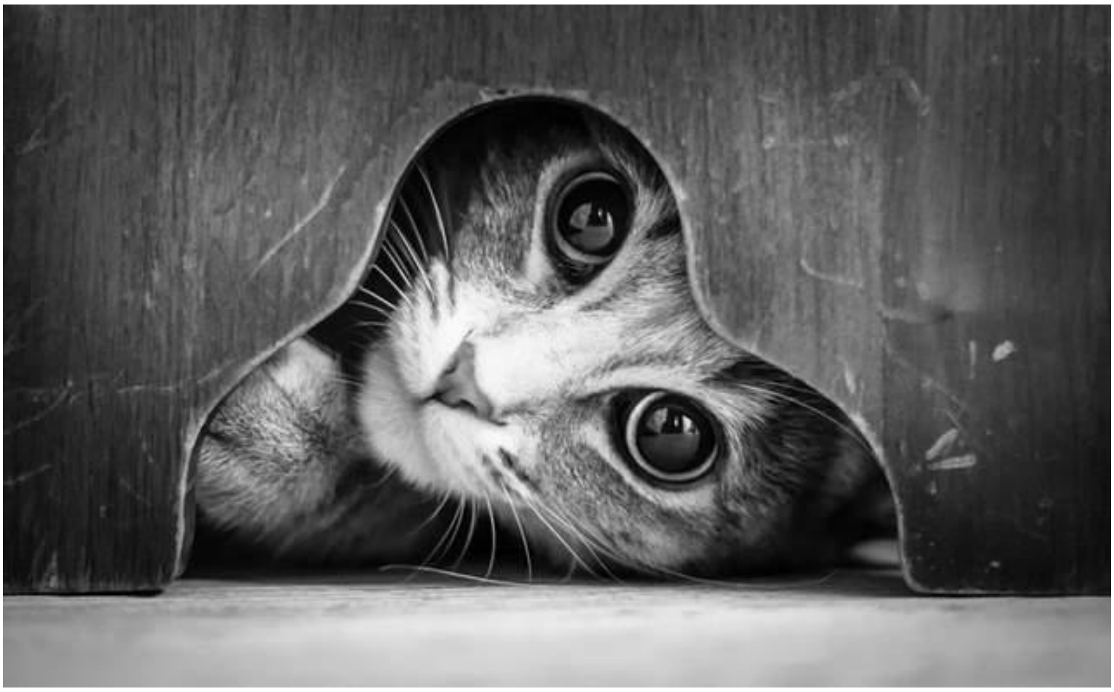
Comment

灰階圖的直方圖均衡化相對簡單，可以直接使用 OpenCV 提供的 cv2.equalizeHist() 來處理，因為它僅針對單通道的亮度資訊進行調整。

然而，對於彩色圖像，不能直接對每個通道獨立應用直方圖均衡化，否則可能會導致色彩失真。因此，一種更合適的方法是 **先將彩色圖像轉換為灰階圖，並在灰階圖上應用直方圖均衡化，以增強亮度與對比度**。接著，將均衡化後的灰階影像視為新的亮度參考，並根據其與原始灰階影像的比例來調整原始彩色圖像的 RGB 值。這樣可以在提升影像細節與對比的同時，保留原始的色彩關係，使影像看起來更自然且不會有色偏的問題。

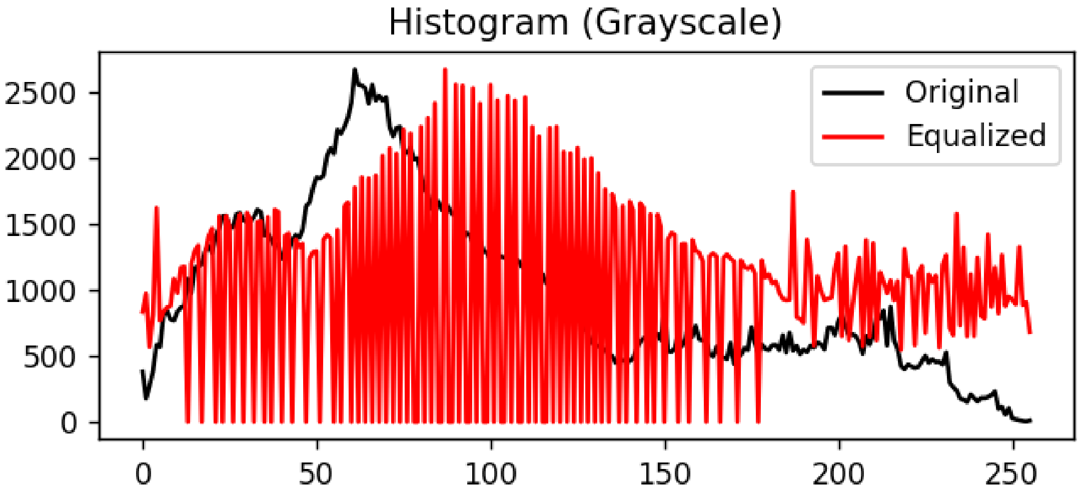
Photos

Origin gray image



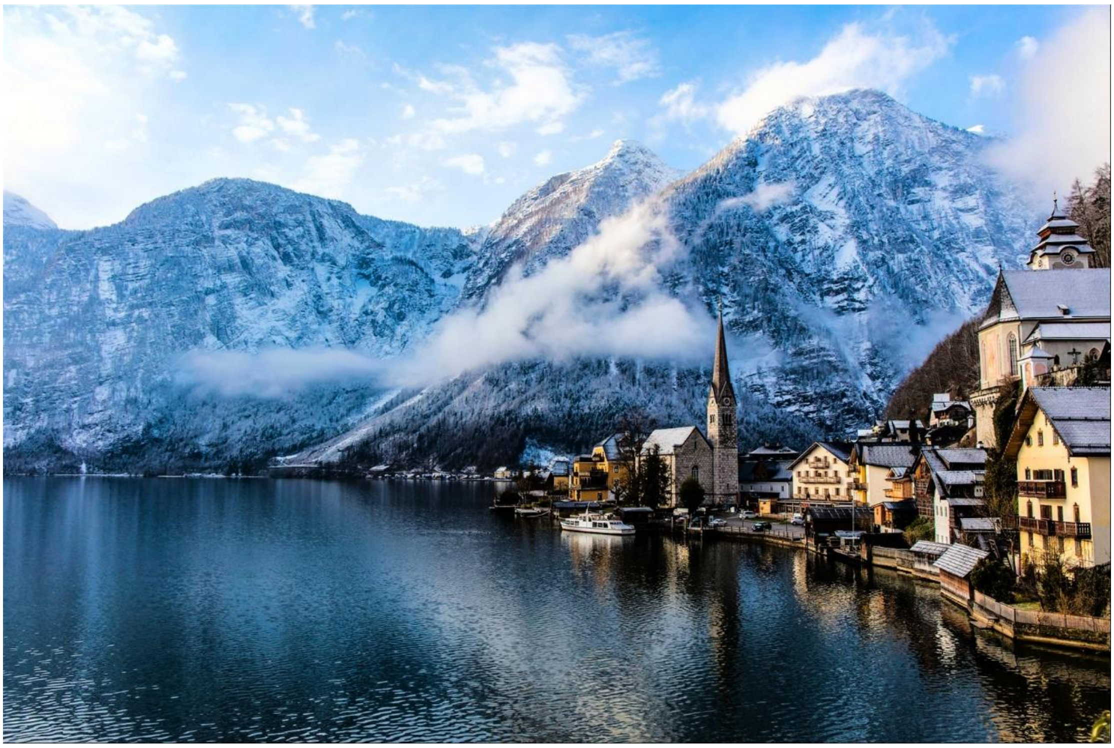
Histogram gray image

Histogram (gray)

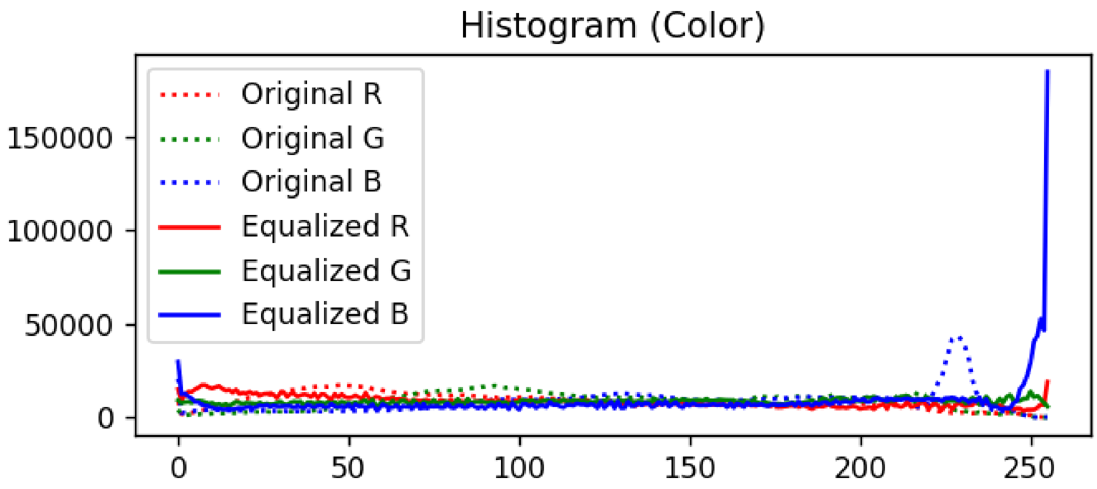


Origin color image

Histogram color image



Histogram (color)



Program

```
import cv2

import numpy as np

import matplotlib.pyplot as plt



gray_image = cv2.imread('cat.jpg', cv2.IMREAD_GRAYSCALE)

color_image = cv2.imread('landscape.jpg')



if gray_image is None or color_image is None:

    print("Error: Image not found!")

    exit()



### 灰階圖

equalized_gray = cv2.equalizeHist(gray_image)



hist_gray_original = cv2.calcHist([gray_image], [0], None, [256], [0, 256])
```

```
hist_gray_equalized = cv2.calcHist([equalized_gray], [0], None, [256], [0, 256])


### 彩圖
gray_from_color = cv2.cvtColor(color_image, cv2.COLOR_BGR2GRAY)
equalized_gray_from_color = cv2.equalizeHist(gray_from_color)


# 避免除 0
gray_from_color = np.maximum(gray_from_color, 1)


# 計算新的 RGB 值 (r’, g’, b’) = (r, g, b) * G’ / G
color_image_float = color_image.astype(np.float32)
equalized_gray_from_color_float = equalized_gray_from_color.astype(np.float32)
gray_from_color_float = gray_from_color.astype(np.float32)


new_color_image = (color_image_float * (equalized_gray_from_color_float[:, :, None] / gray_from_color_float[:, :, None]))
new_color_image = np.clip(new_color_image, 0, 255).astype(np.uint8)


hist_color_original_r = cv2.calcHist([color_image], [2], None, [256], [0, 256])
hist_color_original_g = cv2.calcHist([color_image], [1], None, [256], [0, 256])
hist_color_original_b = cv2.calcHist([color_image], [0], None, [256], [0, 256])


hist_color_equalized_r = cv2.calcHist([new_color_image], [2], None, [256], [0, 256])
hist_color_equalized_g = cv2.calcHist([new_color_image], [1], None, [256], [0, 256])
```

```
hist_color_equalized_b = cv2.calcHist([new_color_image], [0], None, [256], [0, 256])

plt.figure(figsize=(15, 5))

plt.subplot(2, 3, 1)
plt.title("Original Grayscale")
plt.imshow(gray_image, cmap='gray')

plt.subplot(2, 3, 2)
plt.title("Equalized Grayscale")
plt.imshow(equalized_gray, cmap='gray')

plt.subplot(2, 3, 3)
plt.title("Histogram (Grayscale)")
plt.plot(hist_gray_original, color='black', label='Original')
plt.plot(hist_gray_equalized, color='red', label='Equalized')
plt.legend()

plt.subplot(2, 3, 4)
plt.title("Original Color Image")
plt.imshow(cv2.cvtColor(color_image, cv2.COLOR_BGR2RGB))

plt.subplot(2, 3, 5)
plt.title("Equalized Color Image")
plt.imshow(cv2.cvtColor(new_color_image, cv2.COLOR_BGR2RGB))


plt.subplot(2, 3, 6)
plt.title("Histogram (Color)")
plt.plot(hist_color_original_r, color='r', linestyle='dotted', label='Original R')
plt.plot(hist_color_original_g, color='g', linestyle='dotted', label='Original G')
plt.plot(hist_color_original_b, color='b', linestyle='dotted', label='Original B')
plt.plot(hist_color_equalized_r, color='r', label='Equalized R')
plt.plot(hist_color_equalized_g, color='g', label='Equalized G')
plt.plot(hist_color_equalized_b, color='b', label='Equalized B')
plt.legend()


plt.tight_layout()
plt.show()


cv2.imwrite('equalized_cat.jpg', equalized_gray)
cv2.imwrite('equalized_landscape.jpg', new_color_image)
```