import cv2

from colormath.color\_objects import sRGBColor, CMYKColor

from colormath.color\_conversions import convert\_color

import numpy as np

import matplotlib.pyplot as plt

# a

image = cv2.imread('image.jpg', cv2.IMREAD\_GRAYSCALE)

histogram, bins = np.histogram(image.ravel(), 256, [0, 256])

# Plot the histogram

plt.hist(image.ravel(), 256, [0, 256])

plt.xlabel('Intensity Value')

plt.ylabel('Frequency')

plt.title('Image Histogram')

plt.show()

# b

image = cv2.imread('image.jpg')

image\_rgb = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image\_rgb\_color = sRGBColor(\*image\_rgb[0, 0, :])

image\_cmyk\_color = convert\_color(image\_rgb\_color, CMYKColor)

image\_cmyk = np.array(image\_cmyk\_color.get\_value\_tuple(), dtype=np.uint8)

image\_cmyk = cv2.cvtColor(image\_cmyk, cv2.COLOR\_RGB2BGR)

plt.subplot(1, 3, 1)

plt.imshow(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))

plt.title('Original Image')

plt.subplot(1, 3, 2)

plt.imshow(image\_rgb)

plt.title('RGB Image')

plt.subplot(1, 3, 3)

plt.imshow(cv2.cvtColor(image\_cmyk, cv2.COLOR\_BGR2RGB))

plt.title('CMYK Image')

plt.show()

# c

image = cv2.imread('image.jpg')

b, g, r = cv2.split(image)

hist\_b = cv2.calcHist([b], [0], None, [256], [0, 256])

hist\_g = cv2.calcHist([g], [0], None, [256], [0, 256])

hist\_r = cv2.calcHist([r], [0], None, [256], [0, 256])

plt.figure(figsize=(8, 6))

plt.plot(hist\_b, color='b', label='Blue')

plt.plot(hist\_g, color='g', label='Green')

plt.plot(hist\_r, color='r', label='Red')

plt.title('Color Space Histograms')

plt.xlabel('Pixel Intensity')

plt.ylabel('Frequency')

plt.legend()

plt.show()

# d

image = cv2.imread('image.jpg', 0)

equ = cv2.equalizeHist(image)

plt.subplot(1, 2, 1)

plt.imshow(image, cmap='gray')

plt.title('Original Image')

plt.subplot(1, 2, 2)

plt.imshow(equ, cmap='gray')

plt.title('Equalized Image')

plt.show()

# e

image = cv2.imread('image.jpg', 0)

edges = cv2.Canny(image, 100, 200)

plt.subplot(1, 2, 1)

plt.imshow(image, cmap='gray')

plt.title('Original Image')

plt.subplot(1, 2, 2)

plt.imshow(edges, cmap='gray')

plt.title('Edge-Detected Image')

plt.show()

# f

image = cv2.imread('image.jpg')

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

pixels = image.reshape((-1, 3))

pixels = np.float32(pixels)

num\_clusters = 5

max\_iterations = 10

criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, max\_iterations, 1.0)

\_, labels, centers = cv2.kmeans(pixels, num\_clusters, None, criteria, 10, cv2.KMEANS\_RANDOM\_CENTERS)

segmented\_image = labels.reshape(image.shape[:2])

plt.subplot(1, 2, 1)

plt.imshow(image)

plt.title('Original Image')

plt.subplot(1, 2, 2)

plt.imshow(segmented\_image, cmap='nipy\_spectral')

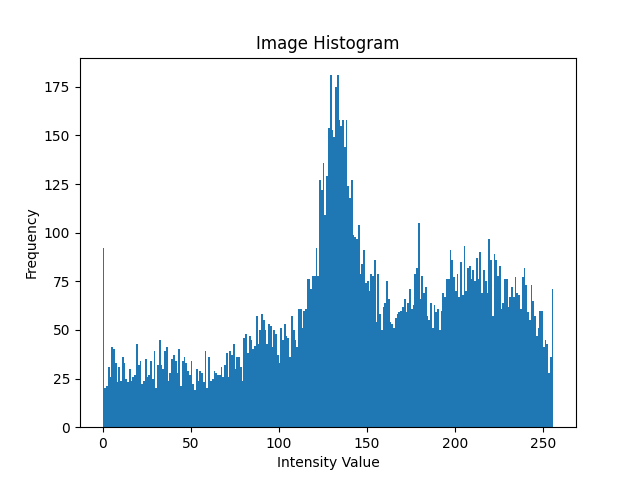
plt.title('Segmented Image')

plt.show()

**img.jpg:**

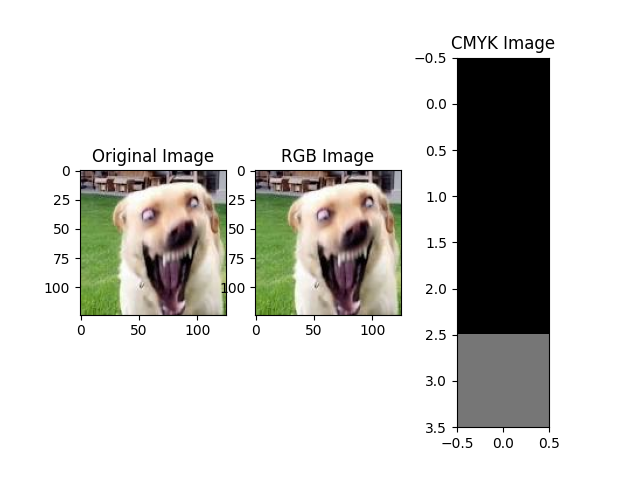


**OUTPUT (a):**

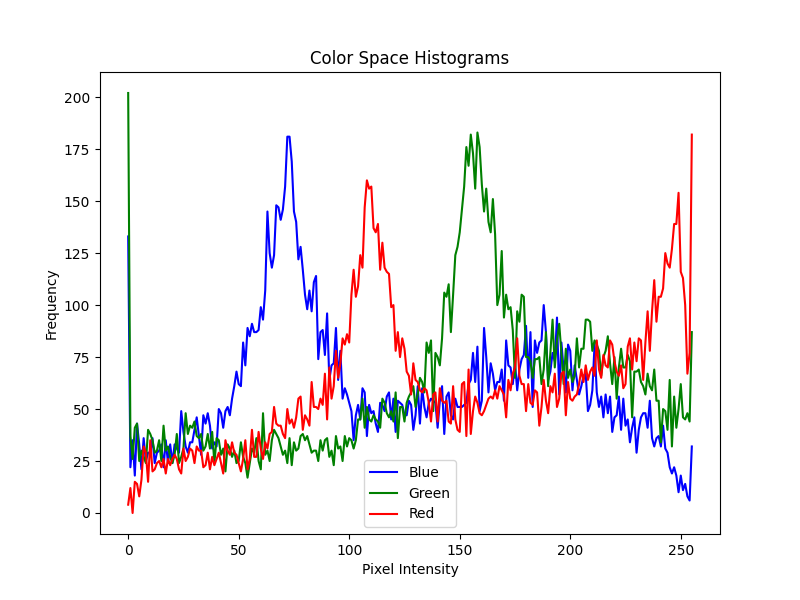


**OUTPUT (b): Original-RGB-CMYK**

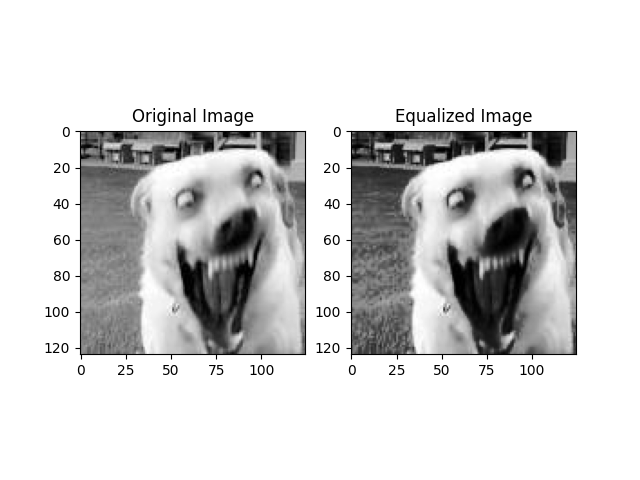
**  No output**

****

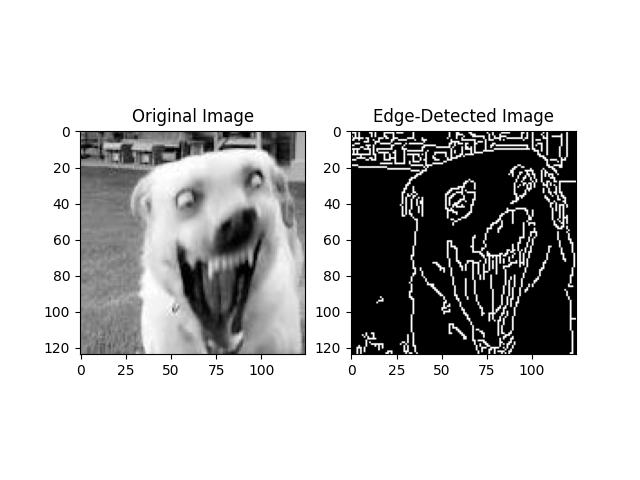
**OUTPUT (c):**

****

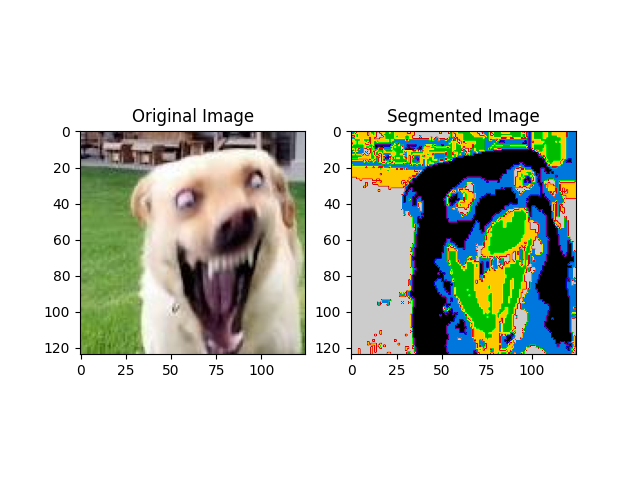
**OUTPUT (d):**

****

**OUTPUT (e):**

****

**OUTPUT (f):**

****