Code For Histogram Equalization

```
import numpy as np
import cv2 as cv

#---------------------------------------RGB-------------------------------------------#

img = cv.imread('./color_img.jpg')
b, g, r = cv.split(img)
L = 256
H, W = b.shape
MN = H * W

blue_hist = cv.calcHist([b], [0], None, [256], [0, 255])
green_hist = cv.calcHist([g], [0], None, [256], [0, 255])
red_hist = cv.calcHist([r], [0], None, [256], [0, 255])

blue_prob = blue_hist / MN
green_prob = green_hist / MN
red_prob = red_hist / MN

cdf_blue = blue_prob
cdf_green = green_prob
cdf_red = red_prob

s_blue = np.zeros(256)
s_green = np.zeros(256)
s_red = np.zeros(256)

s_blue[0] = np.round((L - 1) * cdf_blue[0])
s_green[0] = np.round((L - 1) * cdf_green[0])
s_red[0] = np.round((L - 1) * cdf_red[0])

for i in range (1, 256):
    cdf_blue[i] += cdf_blue[i - 1]
    cdf_green[i] += cdf_green[i - 1]
    cdf_red[i] += cdf_red[i - 1]
    s_blue[i] = np.round((L - 1) * cdf_blue[i])
    s_green[i] = np.round((L - 1) * cdf_green[i])
    s_red[i] = np.round((L - 1) * cdf_red[i])


for i in range (H):
     for j in range (W):
         x = b[i][j]
         y = g[i][j]
         z = r[i][j]
         b[i][j] = s_blue[x]
         g[i][j] = s_green[y]
         r[i][j] = s_red[z]

merge = cv.merge((b, g, r))
```

```
#--------------------------------------HSV-------------------------------------------#

hsv_img = cv.cvtColor(img, cv.COLOR_BGR2HSV)
h, s, v = cv.split(hsv_img)

v_hist = cv.calcHist([v], [0], None, [256], [0, 255])

v_prob = v_hist / MN

s_v = np.zeros(256)
cdf_v = v_prob
s_v[0] = np.round((L - 1) * cdf_v[0])

for i in range (1, 256):
    cdf_v[i] += cdf_v[i - 1]
    s_v[i] = np.round((L - 1) * cdf_v[i])


for i in range (H):
    for j in range (W):
        x = v[i][j]
        v[i][j] = s_v[x]

output = cv.merge((h, s, v))
```

Fig 3.3: Before Equalizing Blue

Fig 3.4: Before Equalizing Green

Fig 3.5: Before Equalizing Red

Fig 3.6: After Equalizing Blue

Fig 3.7: After Equalizing Green

Fig 3.8: After Equalizing Red

Fig 3.9: Input Image

Fig 3.10: Final Equalized Image

Fig 3.11: Before Equalizing V Channel

Fig 3.12: After Equalizing V Channel

Fig 3.13: Input Image in HSV Color Model

Fig 3.14: Final Equalized Image

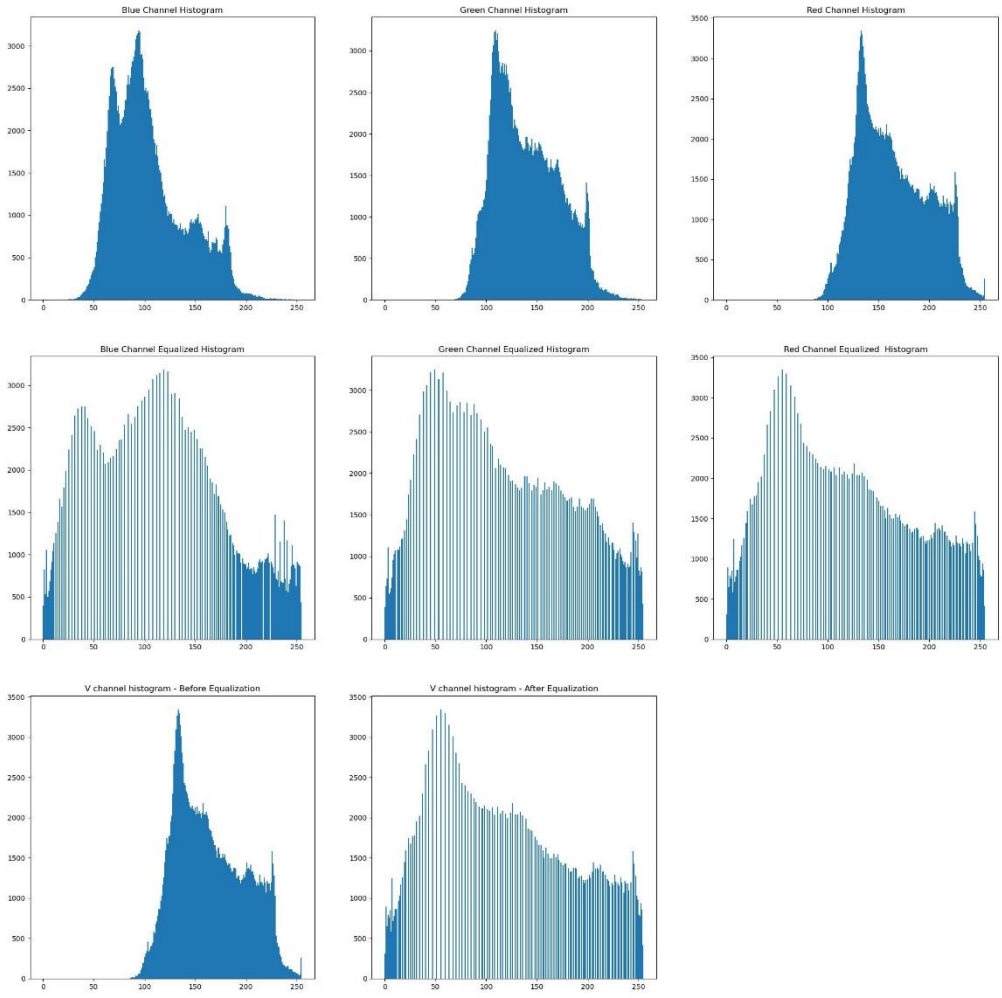

Fig 3.15: Corresponding Histograms

Code For Histogram Matching

```python
import cv2 as cv
import numpy as np
import math
import matplotlib.pyplot as plt

def Gaussian(sigma, mean):
    g = np.zeros(256, dtype = np.float32)
    for i in range (256):
        f = i - mean
        p = -(f ** 2) / (sigma ** 2)
        res = math.exp(p) / (sigma * math.sqrt(2 * math.pi))
        g[i] = res
    return g

sigma1 = 25
mean1 = 100
pdf1 = Gaussian(sigma1, mean1)

sigma2 = 30
mean2 = 180
pdf2 = Gaussian(sigma2, mean2)

final = pdf1 + pdf2
target = np.zeros(256)
target = final / final.sum()

plt.subplot(2, 2, 1)
plt.title('Specified Probability Distribution')
plt.plot(pdf1)
plt.plot(pdf2)
plt.plot(final)
plt.show()


img = cv.imread('./histogram.jpg', cv.IMREAD_GRAYSCALE)
cv.imshow('Input Image', img)

L = 256
M, N = img.shape

hist = cv.calcHist([img],[0],None, [256], [0,255])
pdf = hist / (M * N)
cdf = np.zeros(256, dtype = np.float32)
cdf[0] = pdf[0]
s = pdf

for i in range (1, 256):
    cdf[i] += cdf[i - 1] + pdf[i]
    s[i] = round((L - 1) * cdf[i])

eq_img = np.zeros_like(img)
for i in range (M):
    for j in range(N):
        x = img[i][j]
        eq_img[i][j] = s[x]

plt.subplot(2, 2, 2)
plt.title('Input Histogram')
plt.hist(img.ravel(), 256, [0, 255])

plt.subplot(2, 2, 3)
plt.title('Equalized Histogram')
plt.hist(eq_img.ravel(), 256, [0, 255])
```

```
G = np.zeros(256)
target_cdf = np.zeros(256)
target_cdf[0] = target[0]

for i in range(1, 256):
    target_cdf[i] = target_cdf[i-1] + target[i]
    G[i] = round((L-1) * target_cdf[i])

map = np.zeros(256, dtype = np.int64)
for i in range (256):
    x = np.searchsorted(G, s[i])
    if x > 0 and abs(s[i] - G[x-1]) < abs(G[x] - s[i]):
        x = x-1
    map[int(s[i])] = x

final_image = np.zeros_like(img)
for i in range(eq_img.shape[0]):
    for j in range(eq_img.shape[1]):
        x = eq_img[i][j]
        final_image[i][j] = map[x]

plt.subplot(2,2,4)
plt.title('Matched Histogram')
plt.hist(final_image.ravel(), 256, [0, 255])
plt.show()

cv.imshow('Matched Final Image', final_image)
cv.imwrite('./Matched Final Image.jpg', final_image)

cv.waitKey(0)
cv.destroyAllWindows()
```

Fig 3.16: Input Image

Fig 3.17: Final Output Image After Histogram Matching

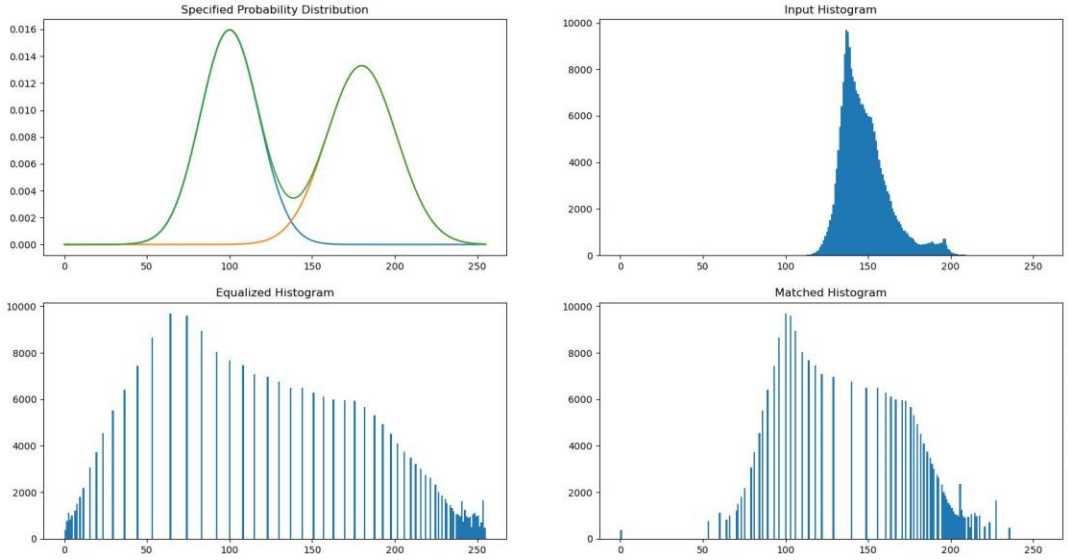

Fig 3.18: Specified Histogram and Final Image Histograms