تمرین اول – بخش عملی

سوال اول – گزارش کار

مهرشاد فلاح اسطلخ‌زیر 401521462

منطق کد:

در این کد عمده تغییرات من در سه تابع بود که به ترتیب به آن‌ها می‌پردازم.

تابع calc_hist:

```
# code here
# first define a function for calculating histogram
# you are free to use libraries
# The last line is the document of this project
def calc_hist(image):
    '''
    you are free to use libraries
    calculate image histogram
    input(s):
      image (ndarray): input image
    output(s):
      hist (ndarray): computed input image histogram
    '''
    hist = np.zeros(256)
    for i in range(256):
        for j in range(256):
            # print(image[i][j])
            hist[image[i][j]] += 1
    return(hist)
```

در این تابع اول یک آرایه 256 تایی متشکل از صفر درست کردم و بعد به ازای هر پیکسلی که در عکس جلو می‌رفتم مقدار رنگ در آن پیکسل را به خانه مرتبطش در آرایه یک واحد اضافه می‌کردم و به این ترتیب هیستوگرام عکس را می‌ساختم.

تابع stretch_hist:

```python
# code here
# define a function (stretch) for stretching(input:image , output: stretched image)

def stretch_hist(image):
    '''
    don't use libraries
    input(s):
        image (ndarray): input image
    output(s):
        output_image (ndarray): enhanced image with histogram stretching
    '''
    output_image = image.copy()

    ########################
    #    your code here    #
    ########################
    histogram = calc_hist(image)
    minimum = 0
    maximum = 255
    for i in range(256):
        if histogram[i] > 0:
            minimum = i
            break
    for i in range(255, -1, -1):
        if histogram[i] > 0:
            maximum = i
            break

    # print(maximum)
    # print(minimum)
    for i in range(256):
        for j in range(256):
            output_image[i][j] = (output_image[i][j] - minimum) / (maximum - minimum) * 255
    return output_image
```

این تابع وظیفه انجام کشش هیستوگرام را داشت و به همین منظور ابتدا هیستوگرام عکس را درست می‌کردم و بعد مینیمم مقدار هیستوگرام در سمت چپ که مساوی صفر نبود و مینیمم مقدار هیستوگرام در سمت راست که مساوی با صفر نبود (یه جورایی اولین مقدار غیر صفر از سمت چپ و راست هیستوگرام) را پیدا می‌کردم و در نهایت فرمول کشش هیستوگرام را بر روی عکس خروجی پیاده کردم.

تابع clip_hist:

```python
# code here
# define a function (Clip) for Clipping(input:image , output: Clipped image)

def clip_hist(image, min_value, max_value):
    '''
    don't use libraries
    input(s):
        image (ndarray): input image
        min_value : min value of the histogram which you wanna clip.
        max_value : max value of the histogram which you wanna clip.
    output(s):
        output_image (ndarray): enhanced image with histogram clipping
    '''
    output_image = image.copy()


    for i in range(256):
        for j in range(256):
            if image[i, j] < min_value:
                output_image[i, j] = 0
            elif image[i, j] > max_value:
                output_image[i, j] = 255
            else:
                output_image[i, j] = (image[i, j] - min_value) / (max_value - min_value) * 255

    return output_image
```

این تابع وظیفه برش هیستوگرام را داشت. در پارامترهای تابع مقادیر مینیمم و ماکسیمم را هم داریم که عمل برش را با استفاده از فرمول بر روی آن‌ها انجام می‌دهیم. به این صورت که مقادیر کوچکتر از مینیمم صفر می‌شوند و مقادیر بزرگتر از ماکسیمم 255 می‌شوند. باقی مقادیر با استفاده از فرمول برش هیستوگرام نگاشت می‌شوند و بدین ترتیب عکس خروجی بدست می‌آید.

بخش خواندن عکس و اجرای توابع:

```python
# define "image1" here

img_path = "./image1.jpg"
img = cv2.imread(img_path)
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
img_gray.shape
```

✓ 0.0s

```
(256, 256)
```

```python
# show the image and its histogram

histogram = calc_hist(img_gray)
plt.plot(histogram)
plt.show()
# It turns out that histogram of gray version of this image is have more values in bright side and bright numbers so we shuld make it darker
```

✓ 0.0s

در این بخش ابتدا عکس را از محلی که در آن قرار دارد می‌خوانیم و چون عکس ما سه کاناله است آن را با استفاده از تابع OpenCV به یک عکس یک کاناله تبدیل می‌کنیم و تابع محاسبه هیستوگرام را بر روی آن صدا زده و هیستوگرام بدست آمده را تبدیل به نمودار می‌کنیم و نمایش می‌دهیم.

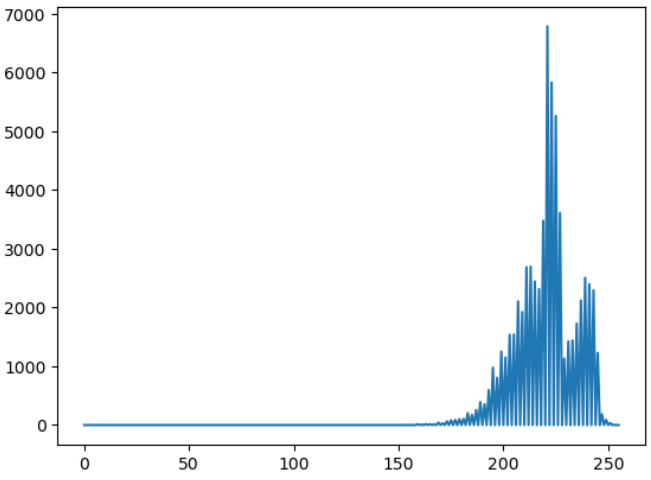
```
# use clip_hist and stretch_hist function to improve quality of the image and show it

stretched_image = stretch_hist(img_gray)
stretched_histogram = calc_hist(stretched_image)
plt.imshow(stretched_image, cmap = 'gray')
plt.show()
plt.plot(stretched_histogram)
plt.show()
clipped_image = clip_hist(img_gray, 175, 240)
clipped_historam = calc_hist(clipped_image)
plt.imshow(clipped_image, cmap='gray')
plt.show()
plt.plot(clipped_historam)
plt.show()
```

✓ 1.1s

بعد از این توابع کشش و برش هیستوگرام را اجرا کرده و هم عکس بدست آمده را نمایش می‌دهیم و هم هیستوگرام مربوط به عکس بدست آمده را.

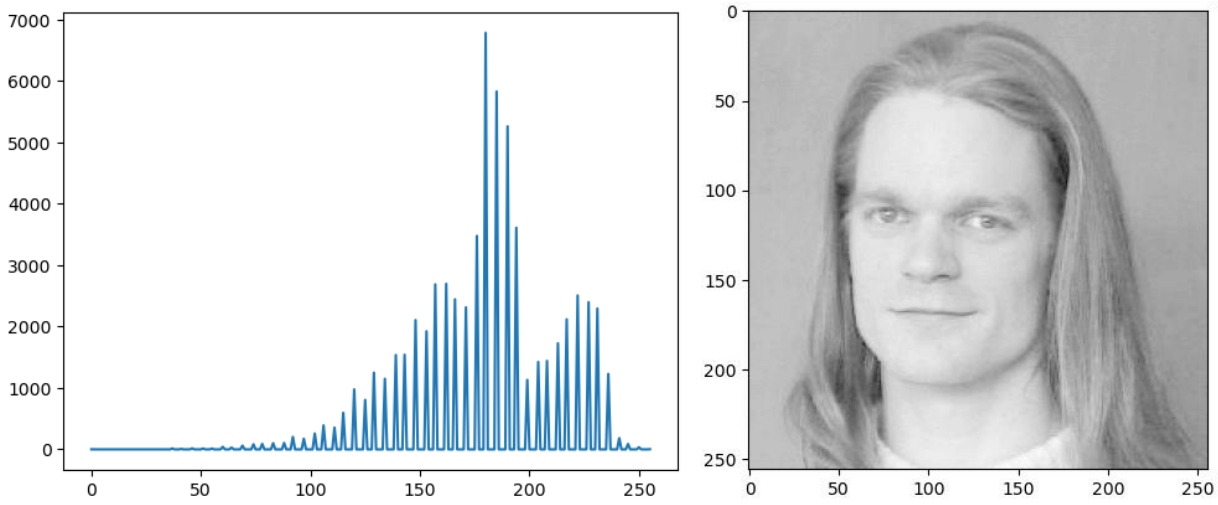
تحلیل نتایج:

عکس و نمودار اولیه:



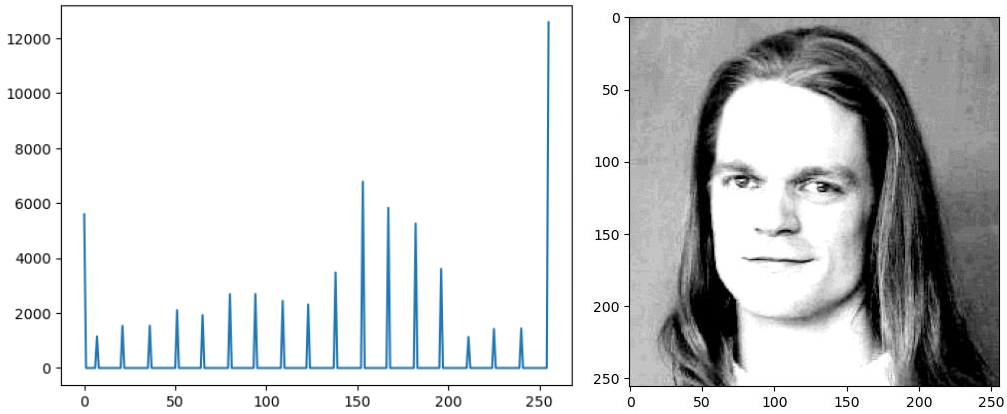
همانطور که در این تصاویر مشخص است. عکس اولیه از کنتراست پایینی برخوردار است و همچنین مقادیر هیستوگرام در رنگ‌های روشن بسیار زیاد است و یعنی عکس روشن است.

کشش هیستوگرام:



با استفاده از کشش هیستوگرام عکس تیره‌تر شده و همچنین کنتراست آن بیشتر شده اما همچنان در مقادیر تیره مقدار هیستوگرام کم است و بیشتر سنگینی نمودار در سمت روشن است چون که مقدار مینیمم نمودار هیستوگرام در عدد 143 بود که به نحوی می‌توان گفت داده پرت محسوب می‌شود.

برش هیستوگرام:



در برش هیستوگرام عکس بدست آمده کنتراست بالاتری دارد و تقریبا از همه رنگ‌ها در آن موجود است و همچنین نمودار هیستوگرام به تابع توزیر یکنواخت شباهت بیشتری دارد از آنجا که این نوع برش به مقادیر مینیمم و ماکسیمم وابستگی زیادی دارد چندین تست مختلف از آن در سلول آخر موجود است که با مقادیر مختلف تست گرفته شده است.

به طور کلی با تحلیل نتایج می‌توان به این نتیجه رسید که برای این عکس برش هیستوگرام با مقدار مینیمم 200 و ماکسیمم 240 مناسب‌تر است و کنتراست تصویر را تا حد خوبی بهتر می‌کند.